install.packages('plotly', repos = 'http://cran.us.r-project.org')

## Installing package into 'C:/Users/Geeta/Documents/R/win-library/3.3'  
## (as 'lib' is unspecified)

## package 'plotly' successfully unpacked and MD5 sums checked  
##   
## The downloaded binary packages are in  
## C:\Users\Geeta\AppData\Local\Temp\Rtmpq4cQWW\downloaded\_packages

library(ggplot2)  
library(GGally)  
  
library(RCurl)

## Loading required package: bitops

library(MASS)  
library(leaps)  
  
install.packages("Boruta", repos = "http://cran.us.r-project.org")

## Installing package into 'C:/Users/Geeta/Documents/R/win-library/3.3'  
## (as 'lib' is unspecified)

## package 'Boruta' successfully unpacked and MD5 sums checked  
##   
## The downloaded binary packages are in  
## C:\Users\Geeta\AppData\Local\Temp\Rtmpq4cQWW\downloaded\_packages

library(Boruta)

## Loading required package: ranger

checkNaFunction <- function(houseData){  
naColumns <- c()  
#checking NA for each columns  
for(i in 1:ncol(houseData)) {  
 #cat(sprintf("Checking NA: %s \n", colnames(houseData)[i]))  
 if(length(which(is.na(houseData[,i]))) > 0){  
 #cat(sprintf("There is NA: %s \n" , colnames(houseData)[i]))  
 naColumns <- c(naColumns, colnames(houseData)[i])  
 }  
}  
return(naColumns)  
}

bucketByColumn <- function(houseData,i){  
minP <- min(as.numeric(houseData[,i]))  
maxP <- max(as.numeric(houseData[,i]))  
rangeP <- range(as.numeric(houseData[,i]))  
rangeP  
cat(sprintf("Min-Max value for: %s , MAX: %d, MIN: %d \n", colnames(houseData)[i], maxP, minP))  
}

findSSEByColName <- function(colName, SSEVals){  
 modlm<-lm(as.formula(paste("log(price)~", paste(c(colName), collapse="+"))),data=train)  
 ## Predicting prices using each Model. we need to take exponent of predict function since it returns log of price.  
 predt<-exp(predict(modlm,newdata=test))  
 SSE<-sum((test$price-predt)^2)  
 SSEVals[[1]]<-c(SSEVals[[1]],colName)  
 SSEVals[[2]]<-c(SSEVals[[2]],SSE)  
 return(SSEVals)  
}

analysis <- function(houseData, i, labels, plotLog, boxLog){  
 #Simple Plot with Price  
 plot(houseData[,i], houseData$price, main = labels[1], xlab = labels[2],ylab = labels[3], col=(c("gold","darkgreen")))  
 #Plot with Log(Price )  
 plot(houseData[,i],log(houseData$price), main = labels[1], xlab =labels[2], ylab = paste('Log of ', labels[3]), col=(c("gold","darkgreen")))  
   
 if(plotLog=='Y'){  
 #Plot with Log(Price ) & Log(independent-variable)  
 plot(log(houseData[,i]+0.5),log(houseData$price), main = labels[1], xlab = paste('Log of ',labels[2]), ylab = paste('Log of ', labels[3]), col=(c("gold","darkgreen")))  
 }  
 #Histogram of independent-variable  
 hist(houseData[,i], main = labels[2], xlab = labels[2], ylab = "Frequency", col=(c("gold","darkgreen")))  
 #BoxPlot of independent-variable  
 boxplot(houseData[,i], main = labels[2], xlab = labels[2], ylab= "Frequency", col=(c("gold","darkgreen")))  
 if(boxLog=='Y'){  
 #BoxPlot of independent-variable & dependent variable  
 boxplot(houseData$price~houseData[,i], main = labels[1], xlab = labels[2], ylab= labels[3], col=(c("gold","darkgreen")))  
 #BoxPlot of independent-variable & dependent variable  
 boxplot(log(houseData$price)~houseData[,i], main = labels[1], xlab = labels[2], ylab= paste('Log of ', labels[3]), col=(c("gold","darkgreen")))  
   
 #BoxPlot of independent-variable & dependent variable  
 boxplot(log(houseData$price)~log(houseData[,i]+0.5), main = labels[1], xlab = paste('Log of ',labels[2]), ylab= paste('Log of ', labels[3]), col=(c("gold","darkgreen")))  
 }  
 #Corelation of independent-variable with Price  
 cor(houseData[,i],houseData$price)  
}

# Data Importing And Cleaning

houseData <- read.csv("file:///G:/Ryerson-BigData/capstone-R/CKME-136/data/kc\_house\_data.csv")  
head(houseData)

## id date price bedrooms bathrooms sqft\_living  
## 1 7129300520 20141013T000000 221900 3 1.00 1180  
## 2 6414100192 20141209T000000 538000 3 2.25 2570  
## 3 5631500400 20150225T000000 180000 2 1.00 770  
## 4 2487200875 20141209T000000 604000 4 3.00 1960  
## 5 1954400510 20150218T000000 510000 3 2.00 1680  
## 6 7237550310 20140512T000000 1225000 4 4.50 5420  
## sqft\_lot floors waterfront view condition grade sqft\_above sqft\_basement  
## 1 5650 1 0 0 3 7 1180 0  
## 2 7242 2 0 0 3 7 2170 400  
## 3 10000 1 0 0 3 6 770 0  
## 4 5000 1 0 0 5 7 1050 910  
## 5 8080 1 0 0 3 8 1680 0  
## 6 101930 1 0 0 3 11 3890 1530  
## yr\_built yr\_renovated zipcode lat long sqft\_living15 sqft\_lot15  
## 1 1955 0 98178 47.5112 -122.257 1340 5650  
## 2 1951 1991 98125 47.7210 -122.319 1690 7639  
## 3 1933 0 98028 47.7379 -122.233 2720 8062  
## 4 1965 0 98136 47.5208 -122.393 1360 5000  
## 5 1987 0 98074 47.6168 -122.045 1800 7503  
## 6 2001 0 98053 47.6561 -122.005 4760 101930

colnames(houseData)

## [1] "id" "date" "price" "bedrooms"   
## [5] "bathrooms" "sqft\_living" "sqft\_lot" "floors"   
## [9] "waterfront" "view" "condition" "grade"   
## [13] "sqft\_above" "sqft\_basement" "yr\_built" "yr\_renovated"   
## [17] "zipcode" "lat" "long" "sqft\_living15"  
## [21] "sqft\_lot15"

naColumns <- checkNaFunction(houseData)  
if(length(naColumns)>0){  
 cat("Found NA Colums:")  
 for(i in 1:length(naColumns)) {  
 cat(sprintf("%s,", colnames(houseData)[i]))  
 }  
}  
  
cat ("Conclusion: Here we conclude that this data does not hold any column with NA.")

## Conclusion: Here we conclude that this data does not hold any column with NA.

##bucketByColumn(houseData,3)

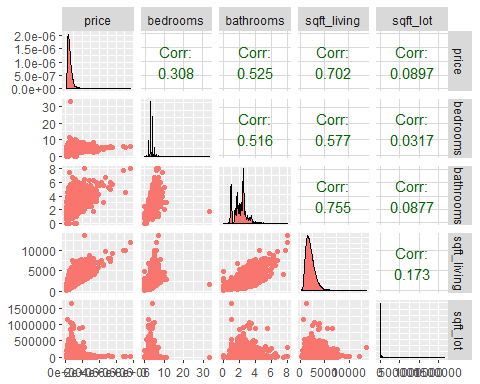
## 

## ###### stepByStep Analysis - START

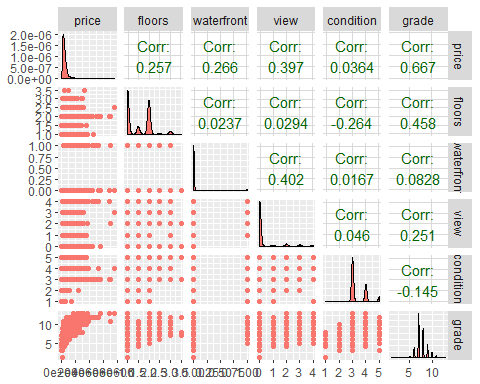
## 

# Price with other attributes

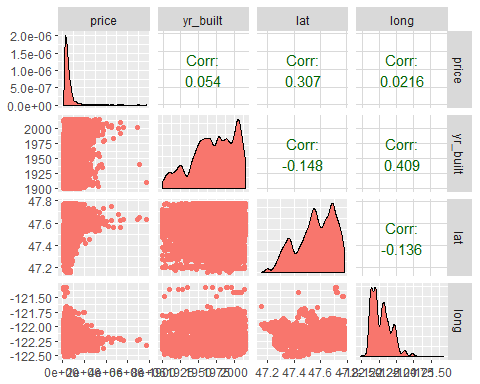
## verify the relationship between price, bedrooms, bathrooms, sqft\_living and sqft lot  
plot1 <- ggpairs(data=houseData, columns=3:7, mapping = aes(color = "dark green"), axisLabels="show")  
plot1



## verify the relationship between price, floors, waterfront, view, condition and grade  
plot2 <- ggpairs(data=houseData, columns=c(3,8:12), mapping = aes(color = "dark green"), axisLabels="show")  
plot2



## verify the relationship between price, yr built, lat and long  
plot3 <- ggpairs(data=houseData, columns=c(3,15,18,19), mapping = aes(color = "dark green"), axisLabels="show")  
plot3



## Correlation among all the variables

#Remove the columns which does not hold any significance in predicing house price  
houseData$date <- NULL  
houseData$id <- NULL  
cor(houseData)

## price bedrooms bathrooms sqft\_living  
## price 1.00000000 0.308349598 0.52513751 0.70203505  
## bedrooms 0.30834960 1.000000000 0.51588364 0.57667069  
## bathrooms 0.52513751 0.515883638 1.00000000 0.75466528  
## sqft\_living 0.70203505 0.576670693 0.75466528 1.00000000  
## sqft\_lot 0.08966086 0.031703243 0.08773966 0.17282566  
## floors 0.25679389 0.175428935 0.50065317 0.35394929  
## waterfront 0.26636943 -0.006582479 0.06374363 0.10381782  
## view 0.39729349 0.079531852 0.18773702 0.28461119  
## condition 0.03636179 0.028472104 -0.12498193 -0.05875259  
## grade 0.66743426 0.356966725 0.66498253 0.76270448  
## sqft\_above 0.60556730 0.477600161 0.68534248 0.87659660  
## sqft\_basement 0.32381602 0.303093375 0.28377003 0.43504297  
## yr\_built 0.05401153 0.154178069 0.50601944 0.31804877  
## yr\_renovated 0.12643379 0.018840823 0.05073898 0.05536293  
## zipcode -0.05320285 -0.152668487 -0.20386627 -0.19943004  
## lat 0.30700348 -0.008931010 0.02457295 0.05252946  
## long 0.02162624 0.129472975 0.22304184 0.24022330  
## sqft\_living15 0.58537890 0.391637524 0.56863429 0.75642026  
## sqft\_lot15 0.08244715 0.029244224 0.08717536 0.18328555  
## sqft\_lot floors waterfront view  
## price 0.089660861 0.256793888 0.266369434 0.397293488  
## bedrooms 0.031703243 0.175428935 -0.006582479 0.079531852  
## bathrooms 0.087739662 0.500653173 0.063743629 0.187737024  
## sqft\_living 0.172825661 0.353949290 0.103817818 0.284611186  
## sqft\_lot 1.000000000 -0.005200991 0.021603683 0.074710106  
## floors -0.005200991 1.000000000 0.023698320 0.029443820  
## waterfront 0.021603683 0.023698320 1.000000000 0.401857351  
## view 0.074710106 0.029443820 0.401857351 1.000000000  
## condition -0.008958250 -0.263767946 0.016653157 0.045989737  
## grade 0.113621124 0.458182514 0.082774914 0.251320585  
## sqft\_above 0.183512281 0.523884710 0.072074592 0.167649344  
## sqft\_basement 0.015286202 -0.245704542 0.080587939 0.276946579  
## yr\_built 0.053080367 0.489319425 -0.026161086 -0.053439851  
## yr\_renovated 0.007643505 0.006338401 0.092884837 0.103917288  
## zipcode -0.129574486 -0.059120642 0.030284728 0.084826917  
## lat -0.085682788 0.049614131 -0.014273776 0.006156732  
## long 0.229520859 0.125419028 -0.041910200 -0.078399712  
## sqft\_living15 0.144608174 0.279885265 0.086463136 0.280439082  
## sqft\_lot15 0.718556752 -0.011269187 0.030703283 0.072574568  
## condition grade sqft\_above sqft\_basement  
## price 0.036361789 0.66743426 0.6055672984 0.32381602  
## bedrooms 0.028472104 0.35696673 0.4776001614 0.30309338  
## bathrooms -0.124981933 0.66498253 0.6853424759 0.28377003  
## sqft\_living -0.058752587 0.76270448 0.8765965987 0.43504297  
## sqft\_lot -0.008958250 0.11362112 0.1835122809 0.01528620  
## floors -0.263767946 0.45818251 0.5238847103 -0.24570454  
## waterfront 0.016653157 0.08277491 0.0720745917 0.08058794  
## view 0.045989737 0.25132058 0.1676493441 0.27694658  
## condition 1.000000000 -0.14467367 -0.1582136164 0.17410491  
## grade -0.144673671 1.00000000 0.7559229376 0.16839182  
## sqft\_above -0.158213616 0.75592294 1.0000000000 -0.05194331  
## sqft\_basement 0.174104914 0.16839182 -0.0519433068 1.00000000  
## yr\_built -0.361416562 0.44696320 0.4238983517 -0.13312410  
## yr\_renovated -0.060617787 0.01441428 0.0232846879 0.07132290  
## zipcode 0.003025524 -0.18486209 -0.2611899765 0.07484461  
## lat -0.014941006 0.11408406 -0.0008164986 0.11053796  
## long -0.106500448 0.19837215 0.3438030175 -0.14476477  
## sqft\_living15 -0.092824268 0.71320209 0.7318702924 0.20035498  
## sqft\_lot15 -0.003405523 0.11924790 0.1940498619 0.01727618  
## yr\_built yr\_renovated zipcode lat  
## price 0.05401153 0.126433793 -0.053202854 0.3070034800  
## bedrooms 0.15417807 0.018840823 -0.152668487 -0.0089310097  
## bathrooms 0.50601944 0.050738978 -0.203866274 0.0245729528  
## sqft\_living 0.31804877 0.055362927 -0.199430043 0.0525294622  
## sqft\_lot 0.05308037 0.007643505 -0.129574486 -0.0856827882  
## floors 0.48931942 0.006338401 -0.059120642 0.0496141310  
## waterfront -0.02616109 0.092884837 0.030284728 -0.0142737756  
## view -0.05343985 0.103917288 0.084826917 0.0061567321  
## condition -0.36141656 -0.060617787 0.003025524 -0.0149410064  
## grade 0.44696320 0.014414281 -0.184862093 0.1140840571  
## sqft\_above 0.42389835 0.023284688 -0.261189977 -0.0008164986  
## sqft\_basement -0.13312410 0.071322902 0.074844608 0.1105379580  
## yr\_built 1.00000000 -0.224873518 -0.346869178 -0.1481224021  
## yr\_renovated -0.22487352 1.000000000 0.064357057 0.0293976092  
## zipcode -0.34686918 0.064357057 1.000000000 0.2670479500  
## lat -0.14812240 0.029397609 0.267047950 1.0000000000  
## long 0.40935620 -0.068372369 -0.564071606 -0.1355117836  
## sqft\_living15 0.32622890 -0.002672555 -0.279032997 0.0488579321  
## sqft\_lot15 0.07095793 0.007853765 -0.147221069 -0.0864188072  
## long sqft\_living15 sqft\_lot15  
## price 0.02162624 0.585378904 0.082447153  
## bedrooms 0.12947298 0.391637524 0.029244224  
## bathrooms 0.22304184 0.568634290 0.087175361  
## sqft\_living 0.24022330 0.756420259 0.183285551  
## sqft\_lot 0.22952086 0.144608174 0.718556752  
## floors 0.12541903 0.279885265 -0.011269187  
## waterfront -0.04191020 0.086463136 0.030703283  
## view -0.07839971 0.280439082 0.072574568  
## condition -0.10650045 -0.092824268 -0.003405523  
## grade 0.19837215 0.713202093 0.119247897  
## sqft\_above 0.34380302 0.731870292 0.194049862  
## sqft\_basement -0.14476477 0.200354983 0.017276181  
## yr\_built 0.40935620 0.326228900 0.070957926  
## yr\_renovated -0.06837237 -0.002672555 0.007853765  
## zipcode -0.56407161 -0.279032997 -0.147221069  
## lat -0.13551178 0.048857932 -0.086418807  
## long 1.00000000 0.334604984 0.254451288  
## sqft\_living15 0.33460498 1.000000000 0.183191749  
## sqft\_lot15 0.25445129 0.183191749 1.000000000

cat ("\nConclusion: sqft\_living , sqft\_above, grade, sqft\_living15, bathrooms have moderate to strong correlation with Price")

##   
## Conclusion: sqft\_living , sqft\_above, grade, sqft\_living15, bathrooms have moderate to strong correlation with Price

cat ("\nConclusion: bathrooms has moderate to strong correlation sqft\_living, sqft\_above, grade, sqft\_living15")

##   
## Conclusion: bathrooms has moderate to strong correlation sqft\_living, sqft\_above, grade, sqft\_living15

cat ("\nConclusion: sqft\_living has moderate to strong correlation sqft\_above, grade, sqft\_living15 and all the above variables studied.")

##   
## Conclusion: sqft\_living has moderate to strong correlation sqft\_above, grade, sqft\_living15 and all the above variables studied.

cat ("\nConclusion: grade has moderate to strong correlation sqft\_above, sqft\_living15 and all the above variables studied.")

##   
## Conclusion: grade has moderate to strong correlation sqft\_above, sqft\_living15 and all the above variables studied.

cat ("\nConclusion: sqft\_above has moderate to strong correlation sqft\_living15 and all the above variables studied.")

##   
## Conclusion: sqft\_above has moderate to strong correlation sqft\_living15 and all the above variables studied.

cat ("\nConclusion: sqft\_basement has moderate correlation with sqft\_living only.")

##   
## Conclusion: sqft\_basement has moderate correlation with sqft\_living only.

cat ("\nConclusion: yr\_built has moderate correlation with bathrooms, floors, grade, sqft\_above only.")

##   
## Conclusion: yr\_built has moderate correlation with bathrooms, floors, grade, sqft\_above only.

cat ("\nConclusion: sqft\_lot15 has strong correlation with sqft\_lot only.")

##   
## Conclusion: sqft\_lot15 has strong correlation with sqft\_lot only.

cat ("\nConclusion: waterfront, view, condition, zipcode, latitude, longitude, yr\_renovated has very weak with other attributes as well as with price.")

##   
## Conclusion: waterfront, view, condition, zipcode, latitude, longitude, yr\_renovated has very weak with other attributes as well as with price.

cat ("\nConclusion: we can see that zip-code has very weak co-orelation -0.053202854, so let us remove it")

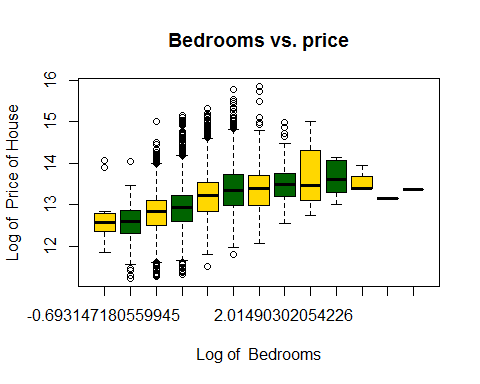
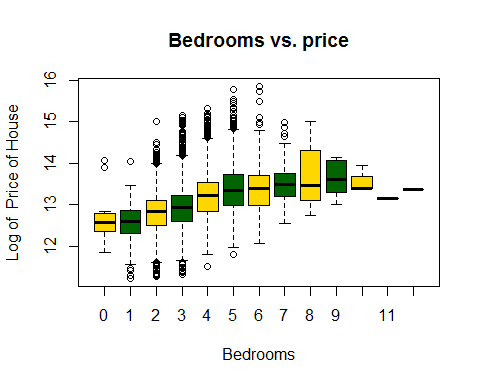
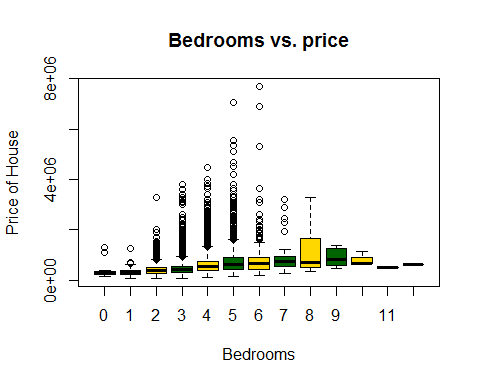
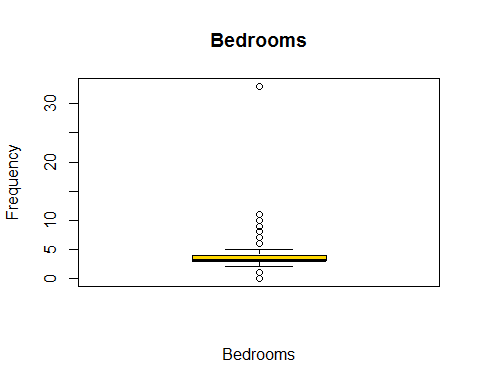
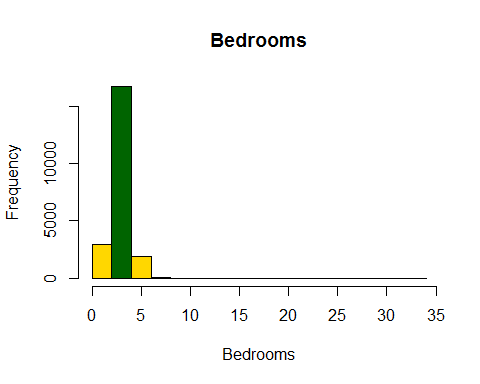
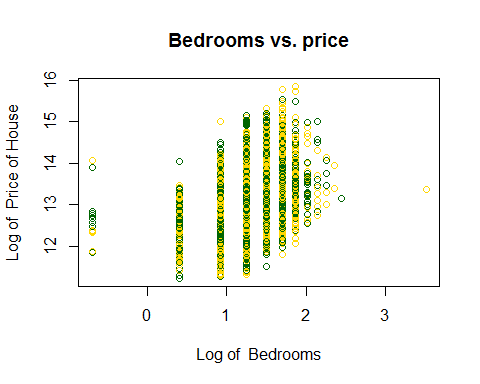
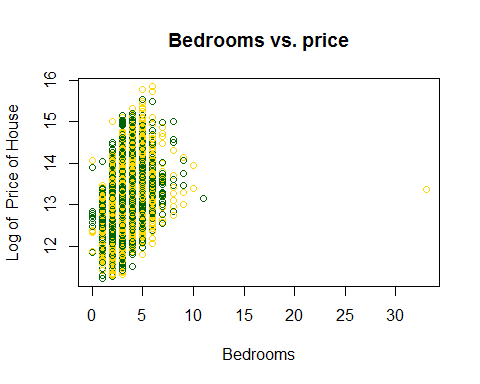
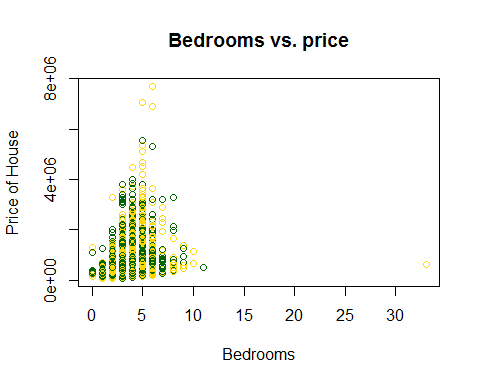
##   
## Conclusion: we can see that zip-code has very weak co-orelation -0.053202854, so let us remove it

houseData$zipcode <- NULL  
 houseData <- read.csv("file:///G:/Ryerson-BigData/capstone-R/CKME-136/data/kc\_house\_data.csv")

# Now Let us do analysis of price with all other variables one by one

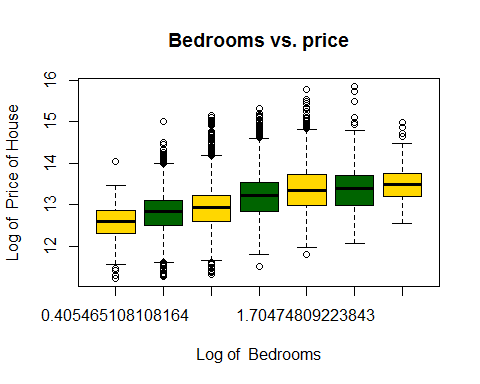
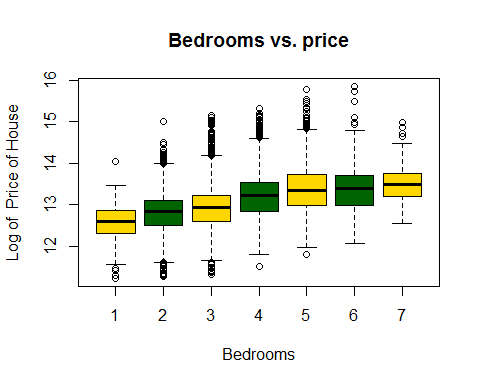
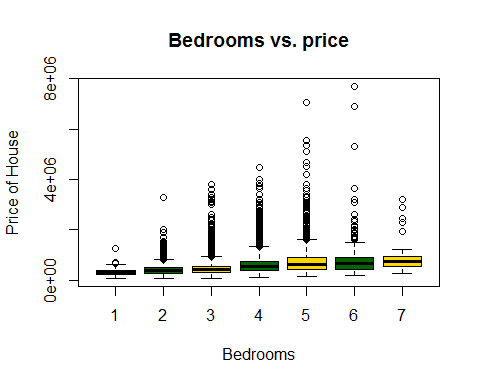
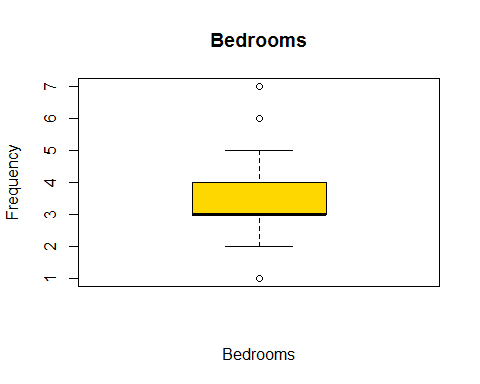
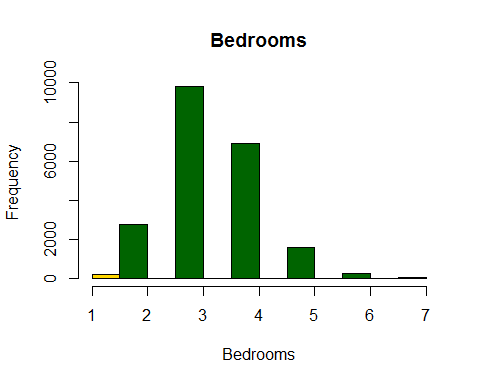
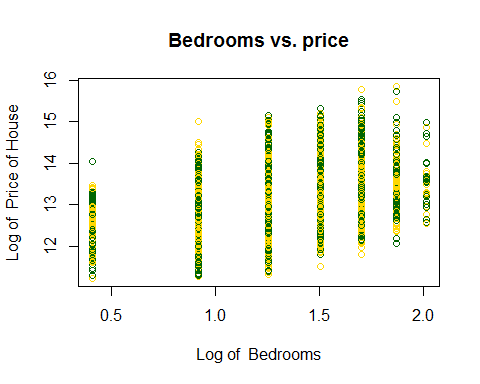
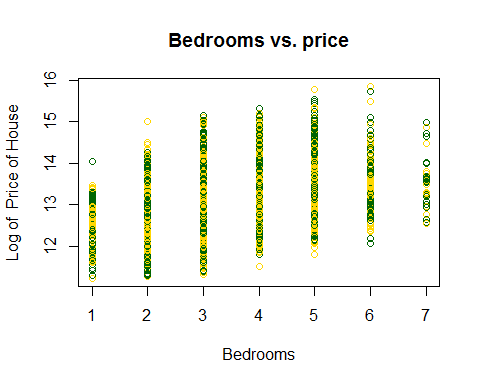
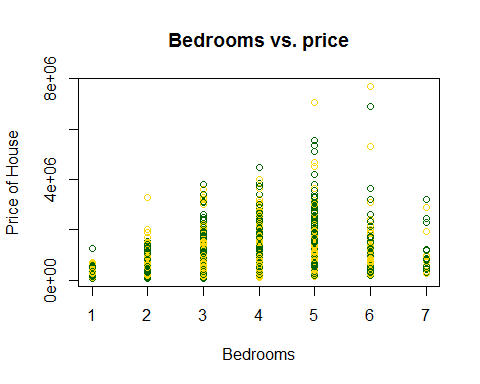
# Bedroom Vs Price analysis

#bucketByColumn(houseData,4)  
analysis(houseData,4,c('Bedrooms vs. price','Bedrooms', 'Price of House'), 'Y', 'Y')



## [1] 0.3083496

#\*\*\*\*\*\*\*Removing the outliers  
#Since more than 7 bedrooms are very rare.Also it's the outlier for my model.  
#I have removed the outlier data.  
houseData<-subset(houseData,bedrooms>=1 & bedrooms<=7)  
#\*\*\*\*\*\*\*Once we removed the outliers, again get the analysis  
analysis(houseData,4,c('Bedrooms vs. price','Bedrooms', 'Price of House'), 'Y', 'Y')



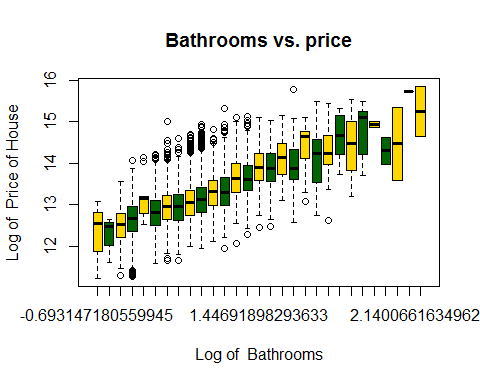
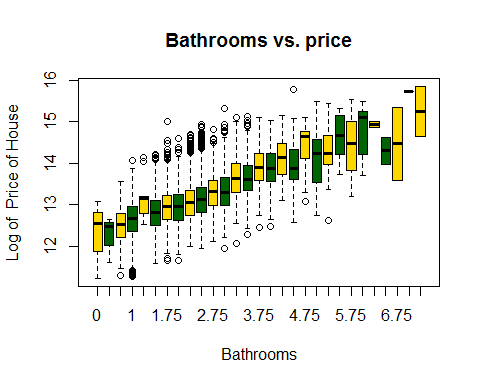
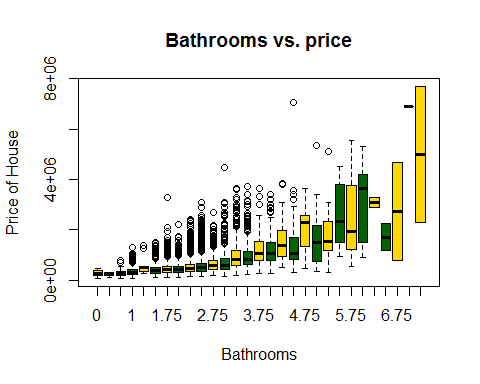
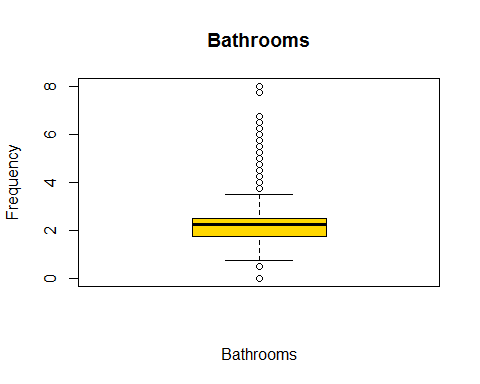
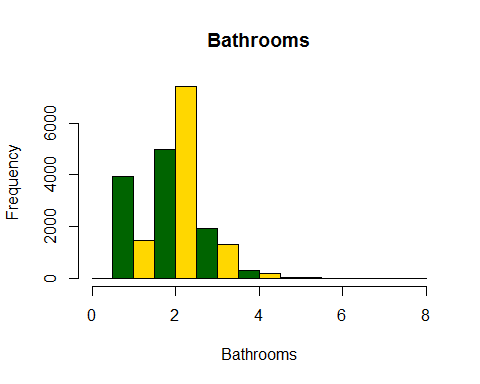
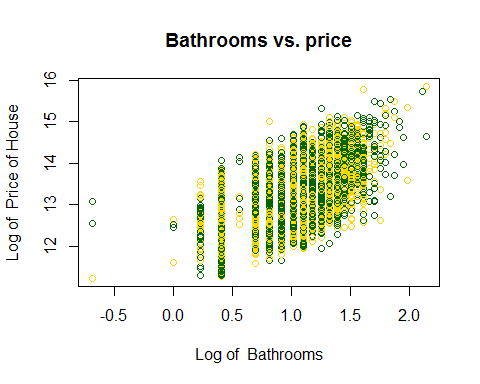
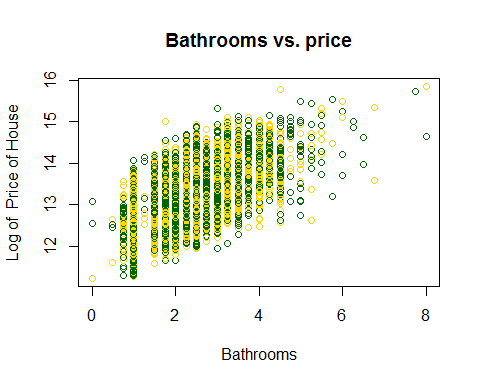
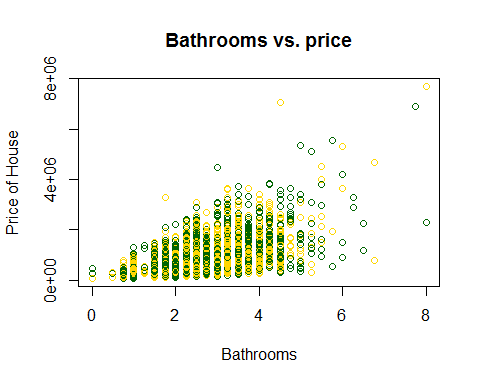
## [1] 0.3156734

#bucketByColumn(houseData,4)  
cat ("Conclusion: here we can found that log of bedroom give better performance than bedroom.")

## Conclusion: here we can found that log of bedroom give better performance than bedroom.

# Bathroom Vs Price analysis

#bucketByColumn(houseData,5)  
analysis(houseData,5,c('Bathrooms vs. price','Bathrooms', 'Price of House'), 'Y', 'Y')

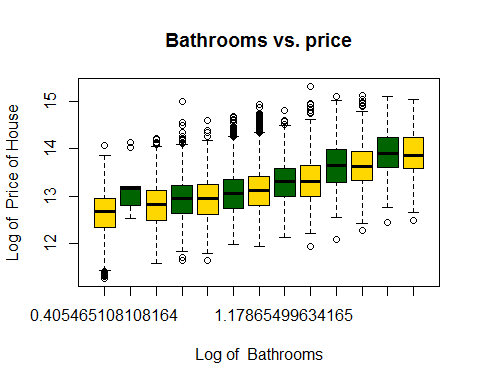
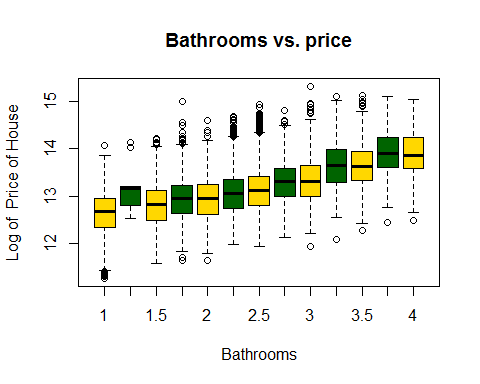
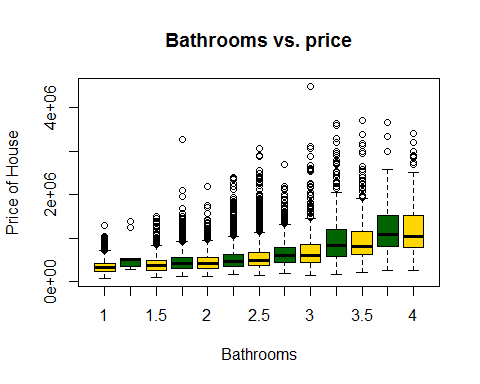
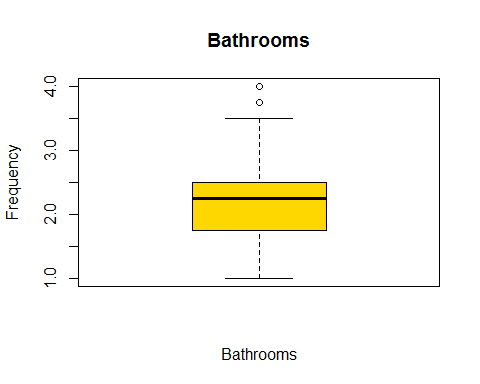
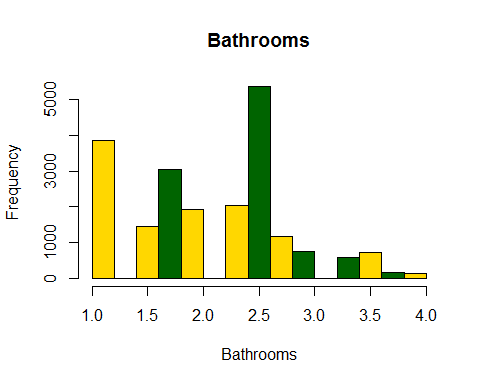
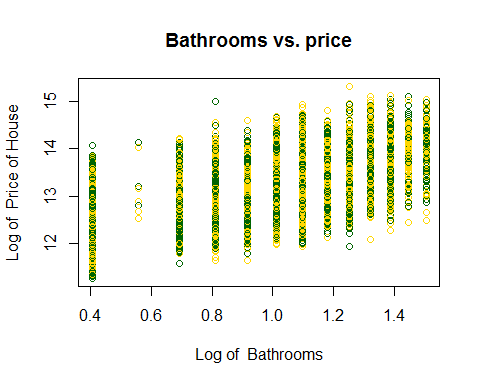
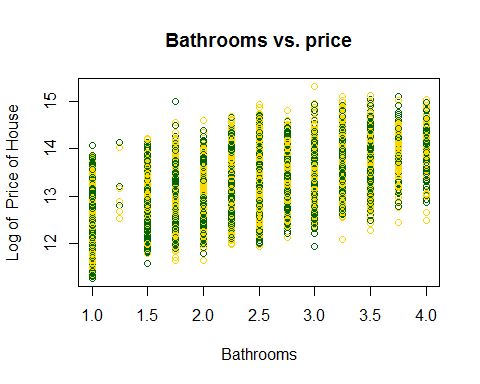
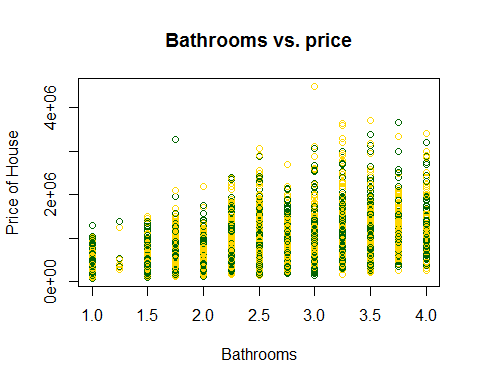


## [1] 0.5259342

cat ("Conclusion: here we found that log of bathrooms give better performance than bathrooms with one expection in when bathroom=7.")

## Conclusion: here we found that log of bathrooms give better performance than bathrooms with one expection in when bathroom=7.

#\*\*\*\*\*\*\*Removing the outliers  
#More than 4 bathrooms are very rare in this data.So I am removing it.  
houseData<-subset(houseData,bathrooms>=1 & bathrooms<=4)  
analysis(houseData,5,c('Bathrooms vs. price','Bathrooms', 'Price of House'), 'Y', 'Y')

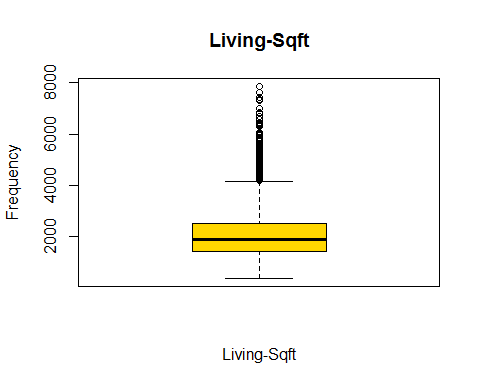
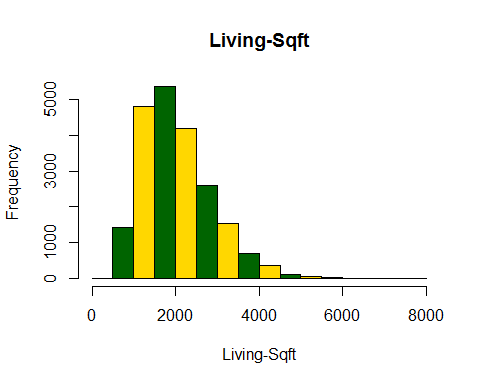
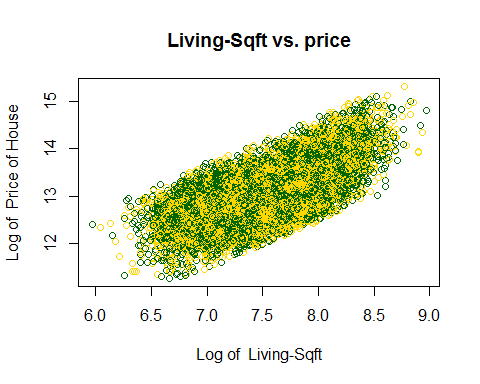
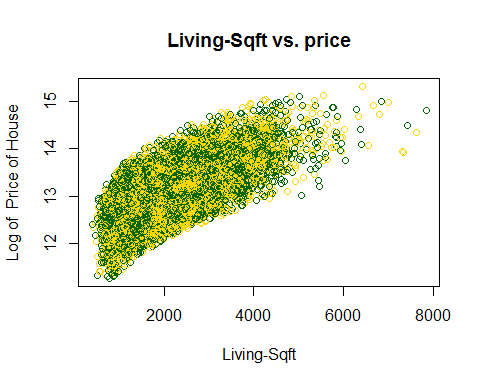
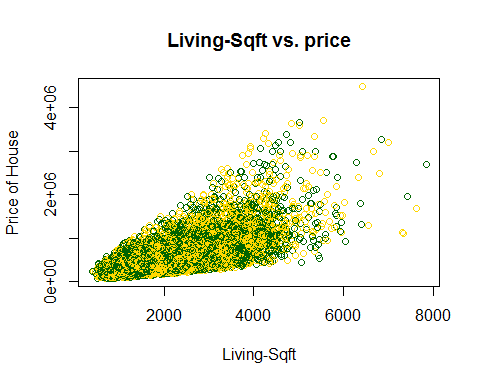


## [1] 0.475159

#bucketByColumn(houseData,5)

# SQFT Living Vs Price analysis

#bucketByColumn(houseData,6)  
analysis(houseData,6,c('Living-Sqft vs. price','Living-Sqft', 'Price of House'), 'Y', 'N')

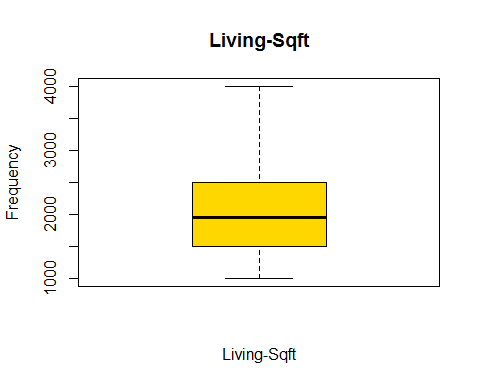
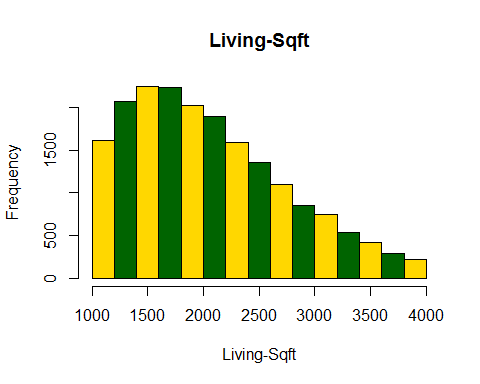
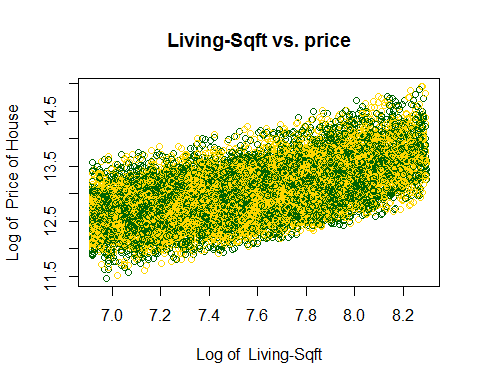
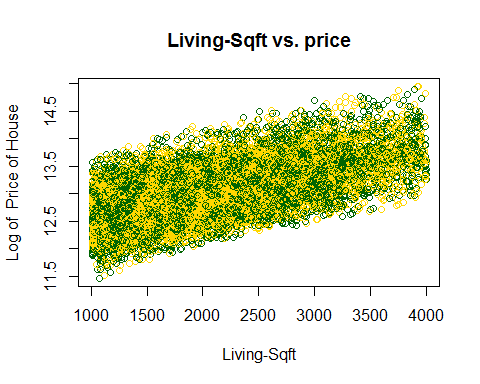
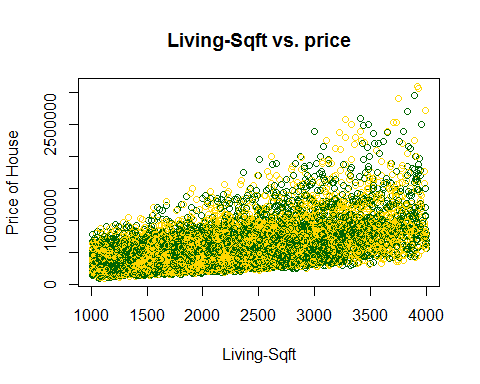


## [1] 0.6701029

cat ("Conclusion: There is Strong correlation between sqft\_living and price as sqft\_living increases, price increases as well.")

## Conclusion: There is Strong correlation between sqft\_living and price as sqft\_living increases, price increases as well.

#\*\*\*\*\*\*\*Removing the outliers  
houseData<-subset(houseData,sqft\_living >1000 & sqft\_living<=4000)  
analysis(houseData,6,c('Living-Sqft vs. price','Living-Sqft', 'Price of House'), 'Y', 'N')

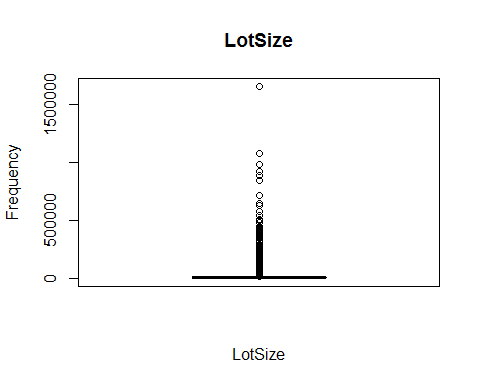
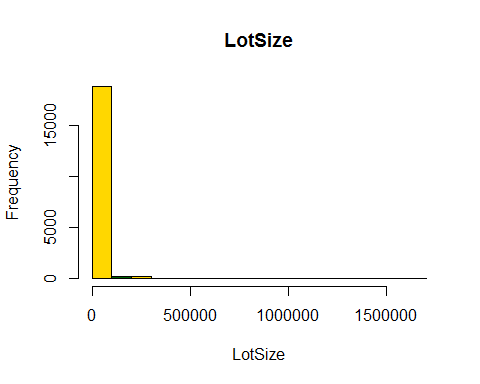
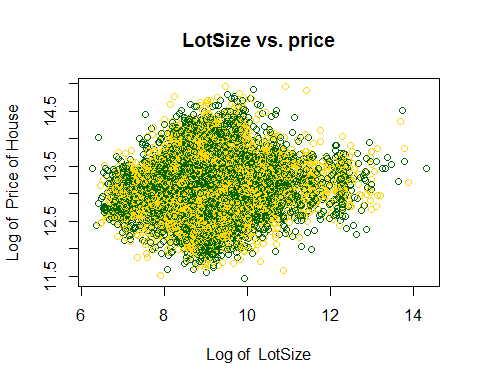
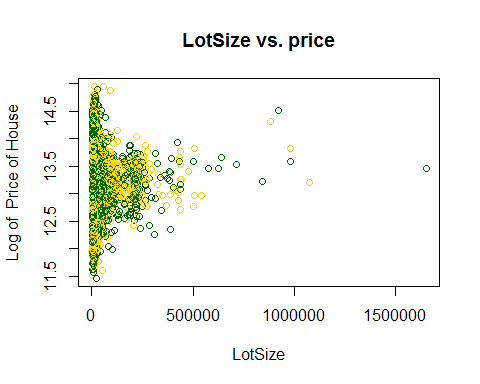
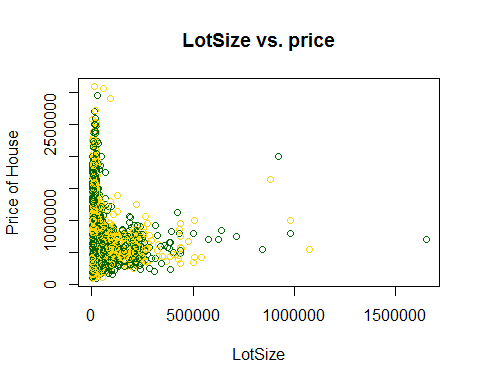


## [1] 0.5938015

#bucketByColumn(houseData,6)

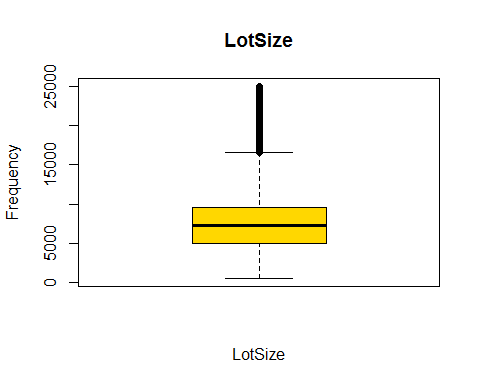
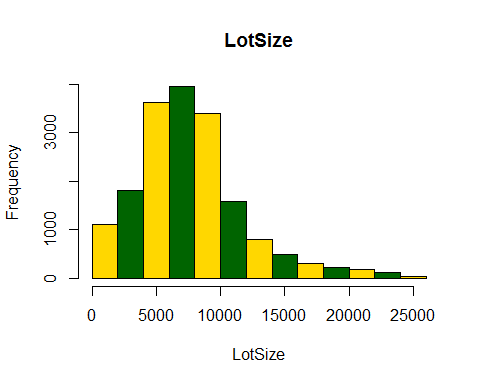
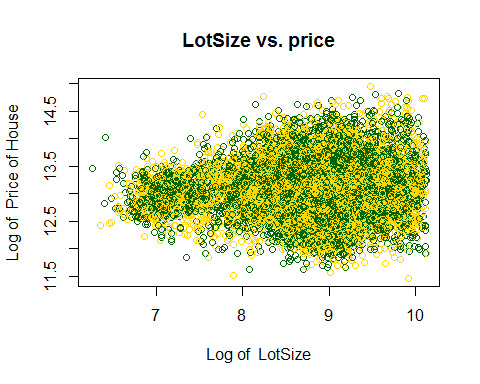
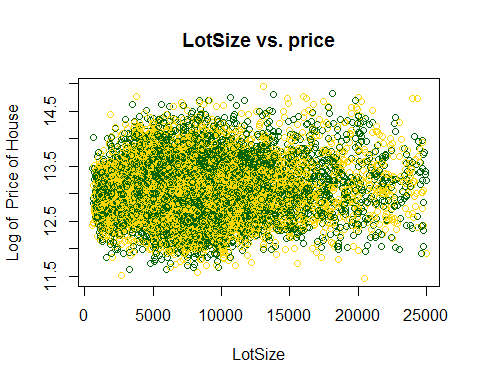
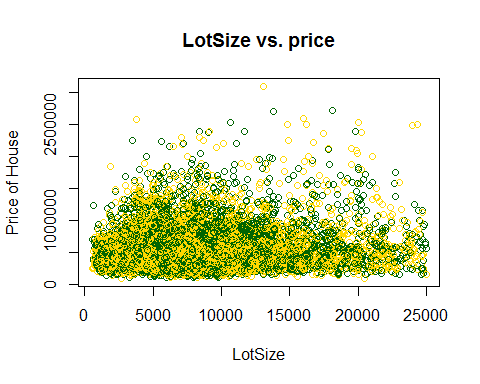
## SQFT\_LOT Vs Price analysis

#bucketByColumn(houseData,7)  
analysis(houseData,7,c('LotSize vs. price','LotSize', 'Price of House'), 'Y', 'N')



## [1] 0.06105332

#\*\*\*\*\*\*\*Removing the outliers  
houseData<-subset(houseData,sqft\_lot>=0 & sqft\_lot<=25000)  
analysis(houseData,7,c('LotSize vs. price','LotSize', 'Price of House'), 'Y', 'N')

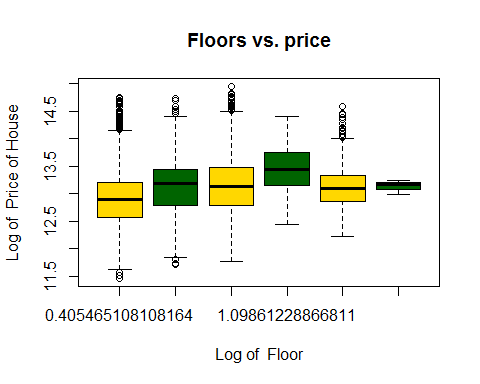
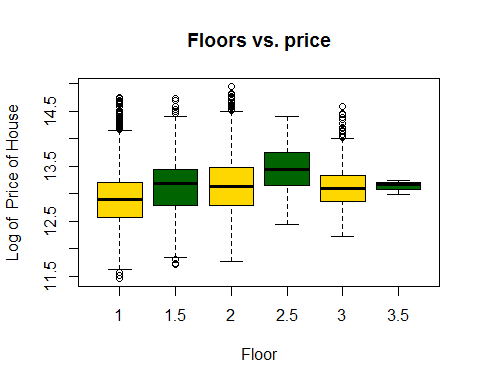
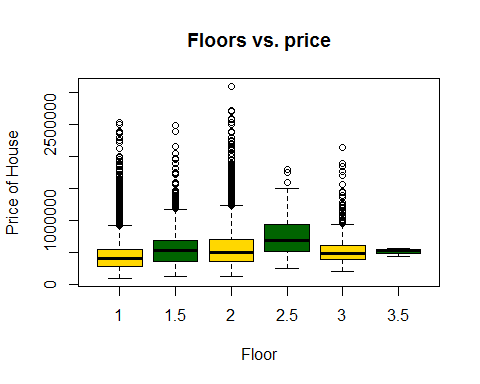
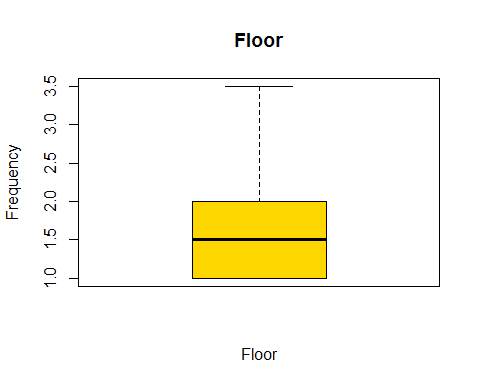
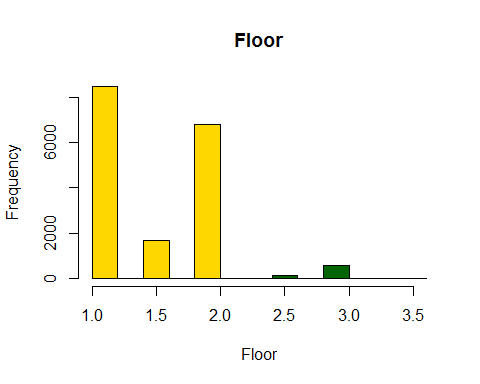
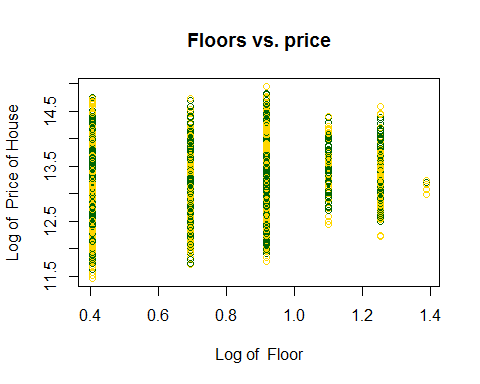
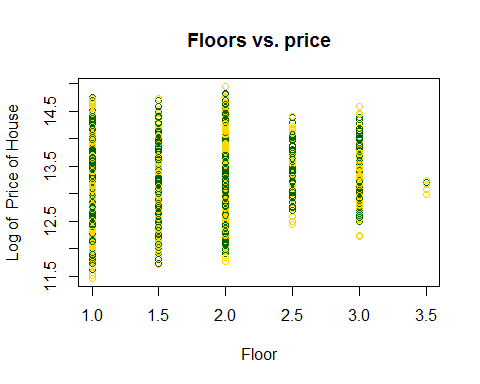
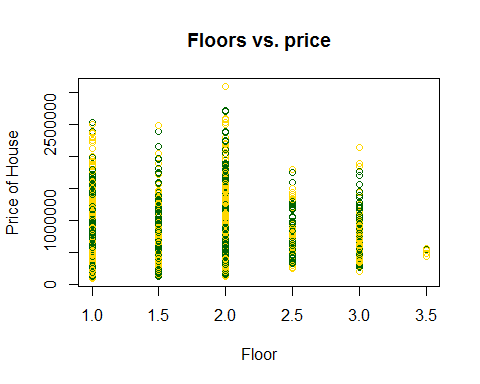


## [1] 0.06714415

#bucketByColumn(houseData,7)

## FLOOR Vs Price analysis

analysis(houseData,8,c('Floors vs. price','Floor', 'Price of House'), 'Y', 'Y')

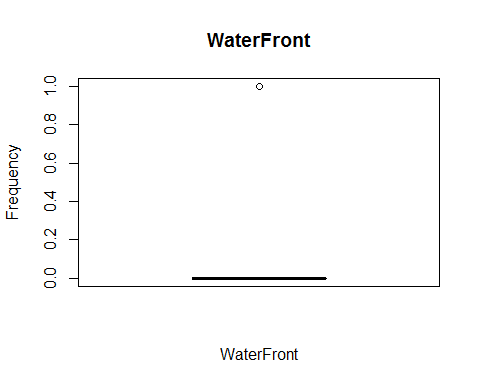
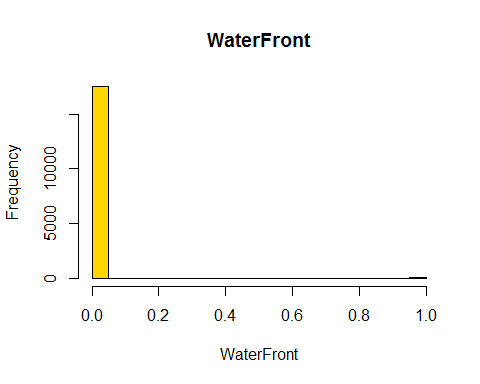
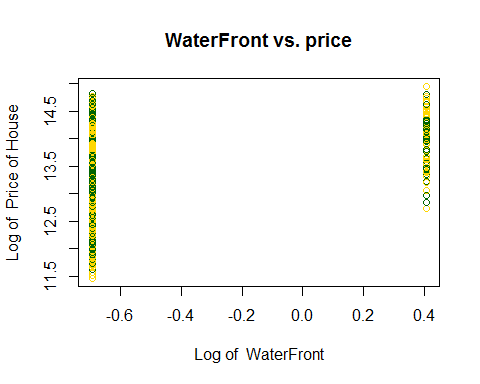
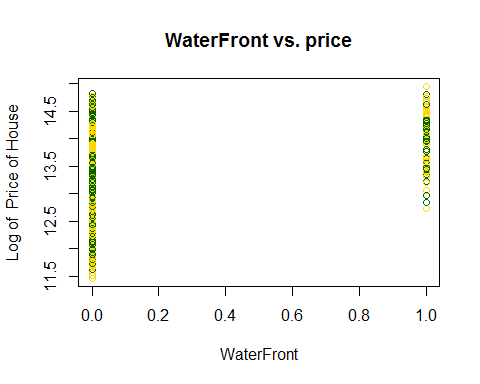
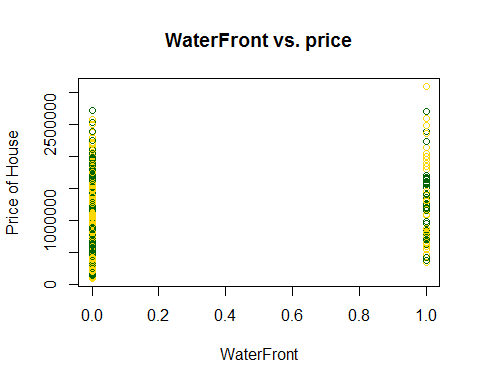


## [1] 0.2072373

##bucketByColumn(houseData,8)

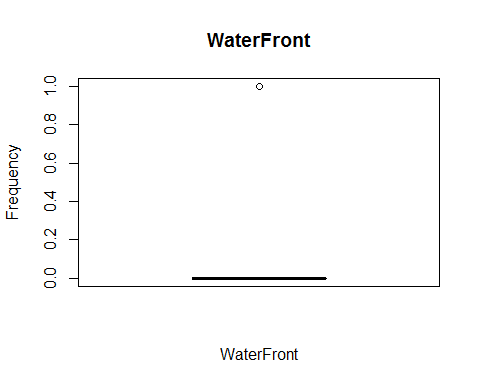
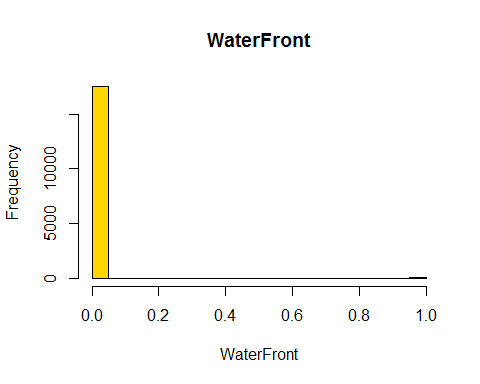
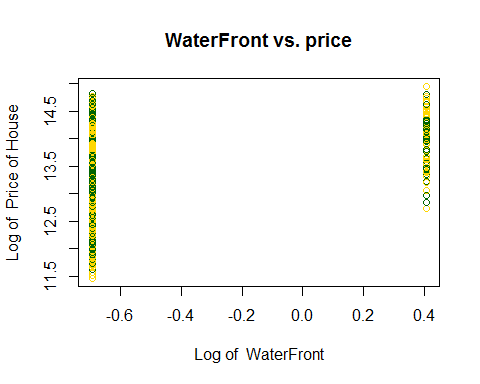
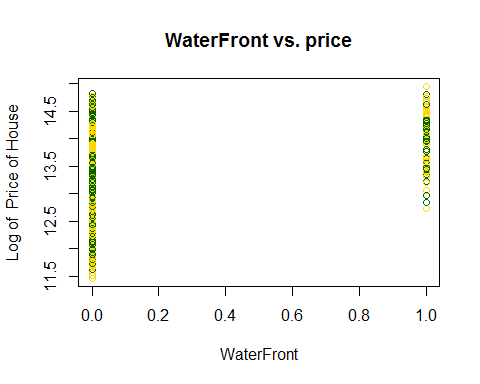
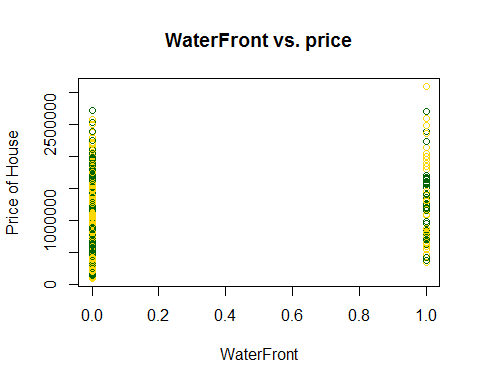
## SQFT\_LOT Vs Price analysis

#bucketByColumn(houseData,9)  
analysis(houseData,9,c('WaterFront vs. price','WaterFront', 'Price of House'), 'Y', 'N')



## [1] 0.209102

#\*\*\*\*\*\*\*Removing the outliers  
houseData<-subset(houseData,sqft\_lot>=0 & sqft\_lot<=25000)  
analysis(houseData,9,c('WaterFront vs. price','WaterFront', 'Price of House'), 'Y', 'N')

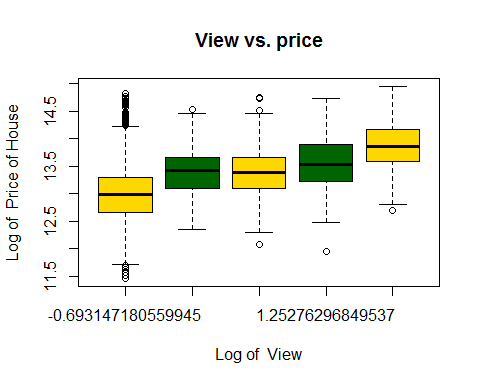
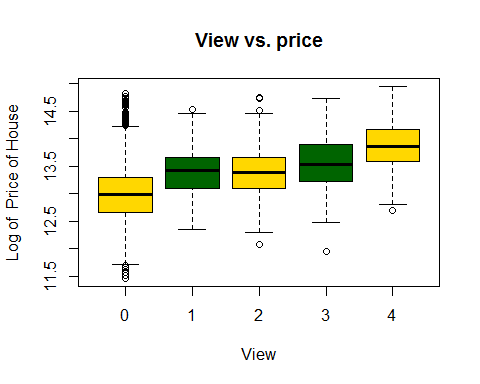
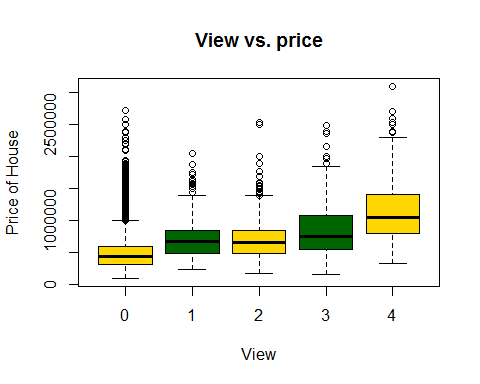
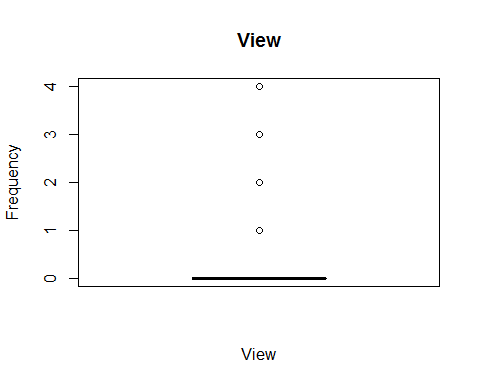
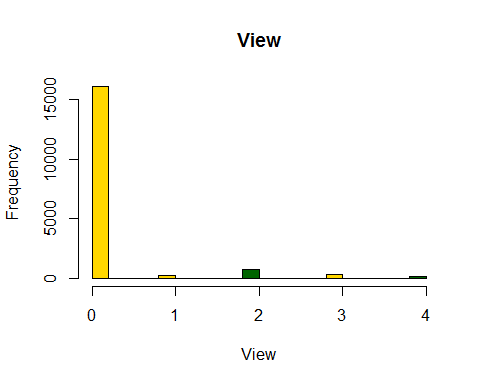
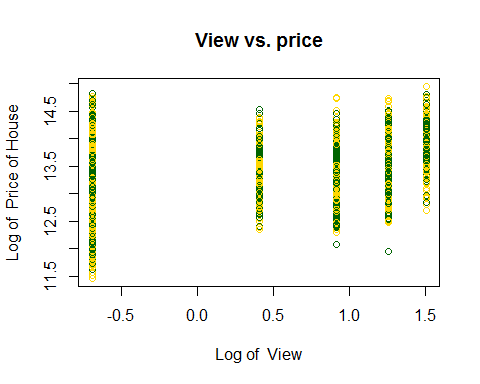
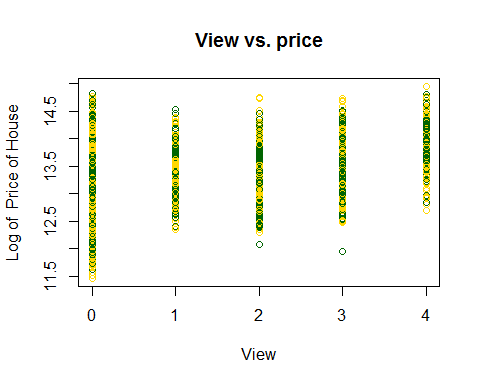
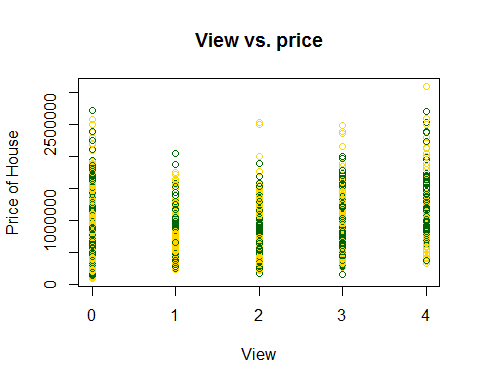


## [1] 0.209102

#bucketByColumn(houseData,9)

## View Vs Price analysis

#bucketByColumn(houseData,10)  
analysis(houseData,10,c('View vs. price','View', 'Price of House'), 'Y', 'Y')

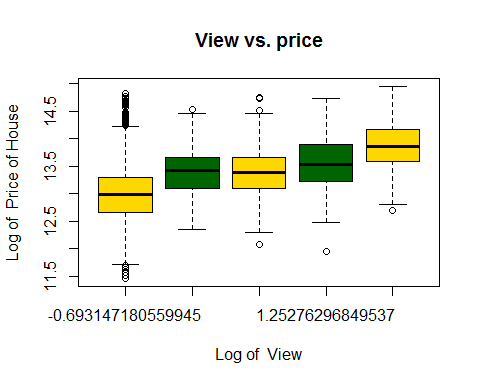
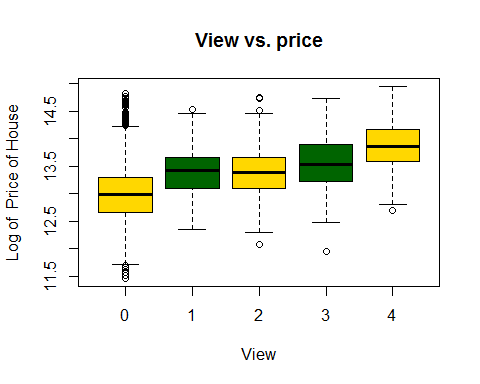
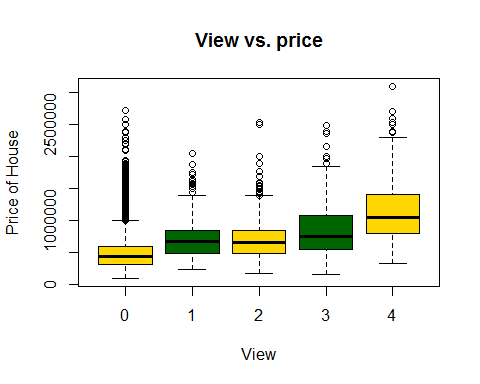
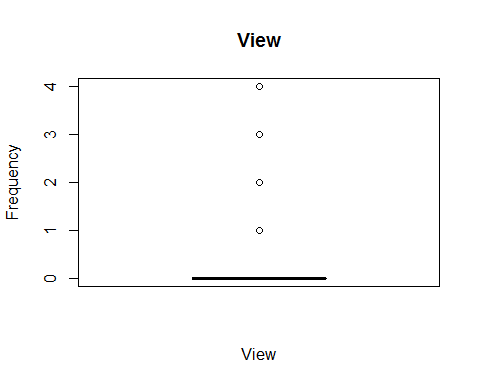
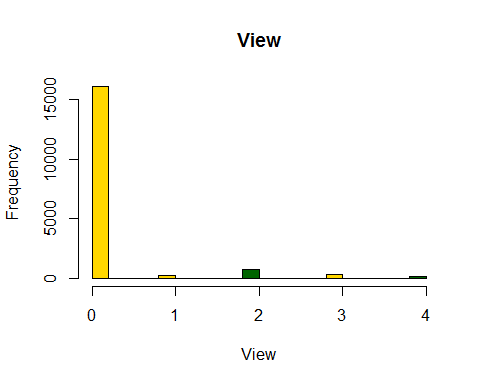
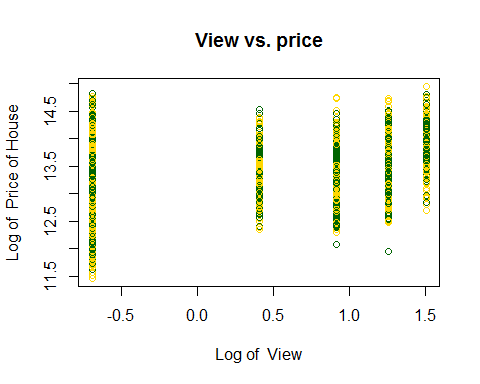
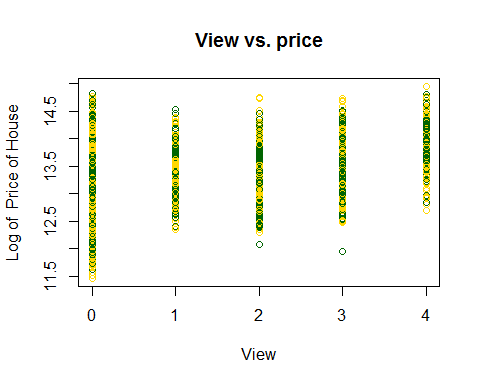
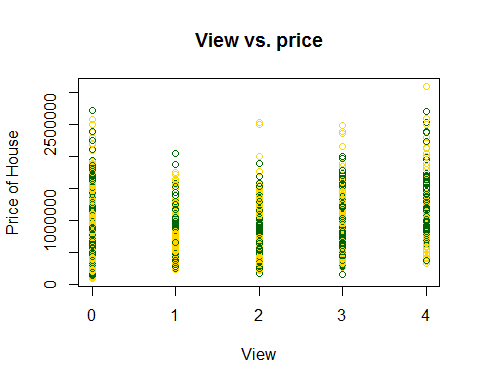


## [1] 0.3578146

cat ("Conclusion: Here we can se that as view increases price also increases as well")

## Conclusion: Here we can se that as view increases price also increases as well

#\*\*\*\*\*\*\*Removing the outliers  
analysis(houseData,10,c('View vs. price','View', 'Price of House'), 'Y', 'Y')

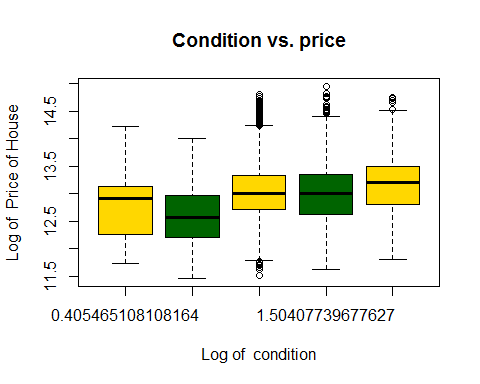
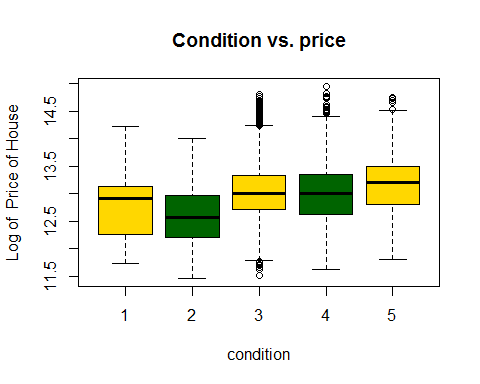
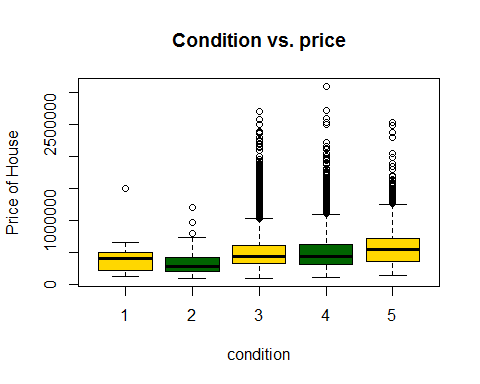
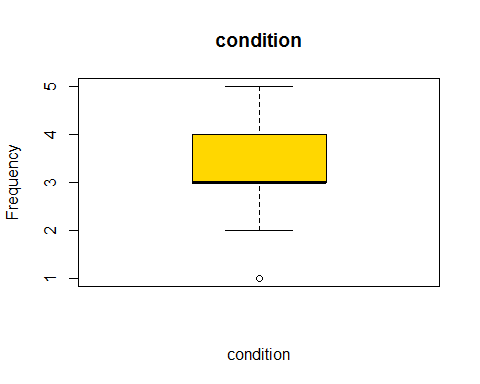
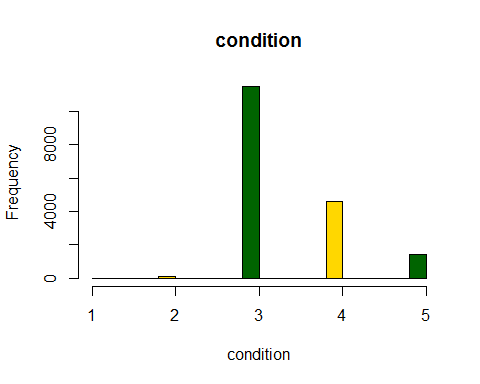
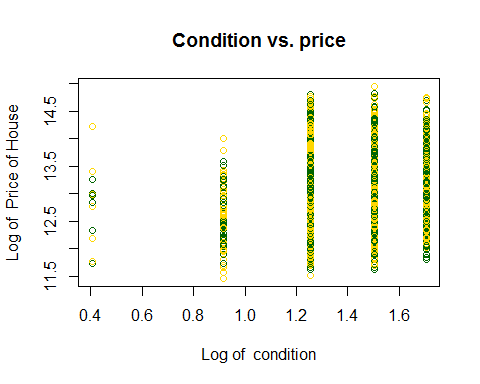
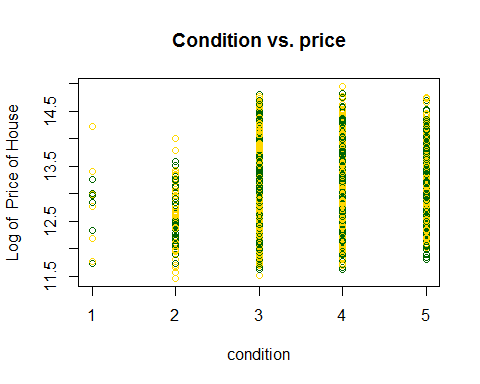
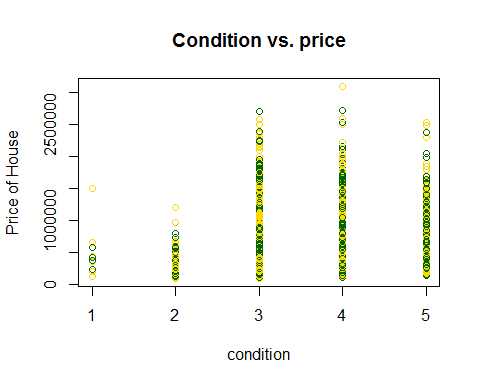


## [1] 0.3578146

#bucketByColumn(houseData,10)

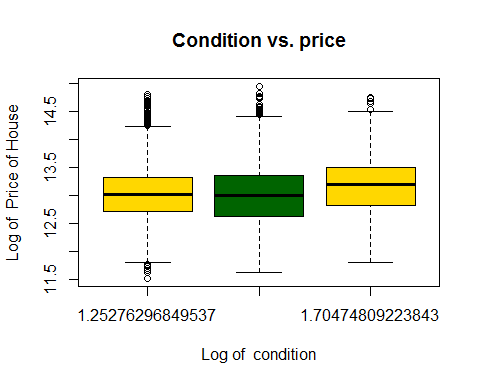
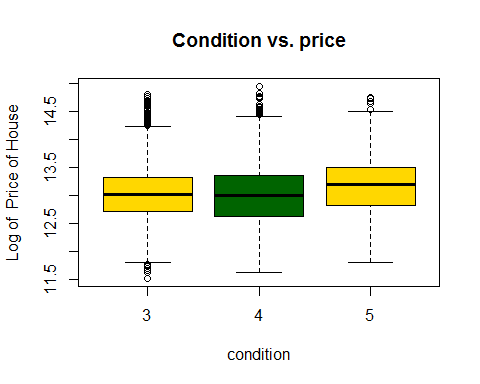
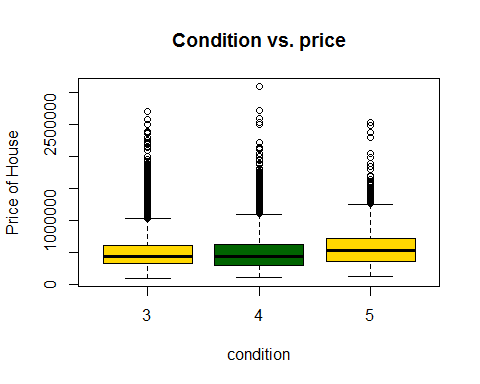
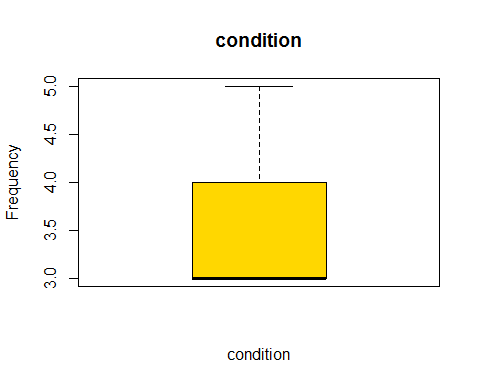
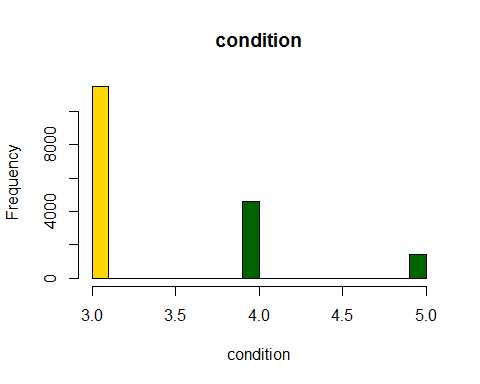
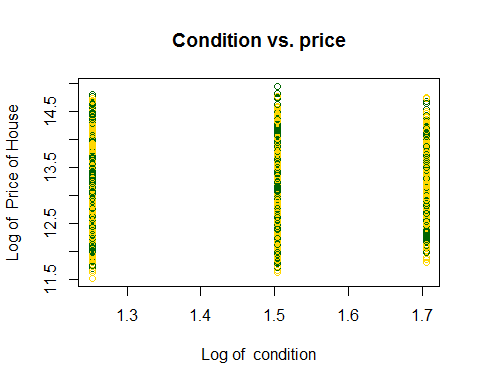
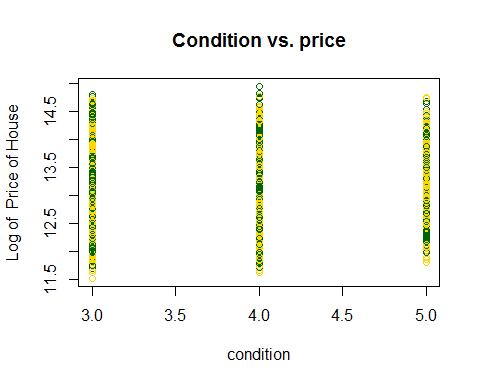
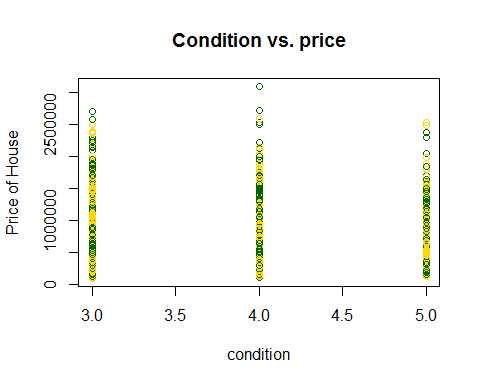
## CONDITION Vs Price analysis

#bucketByColumn(houseData,11)  
analysis(houseData,11,c('Condition vs. price','condition', 'Price of House'), 'Y', 'Y')



## [1] 0.07301379

#\*\*\*\*\*\*\*Removing the outliers  
houseData<-subset(houseData,condition>=3& condition<=5)  
analysis(houseData,11,c('Condition vs. price','condition', 'Price of House'), 'Y', 'Y')

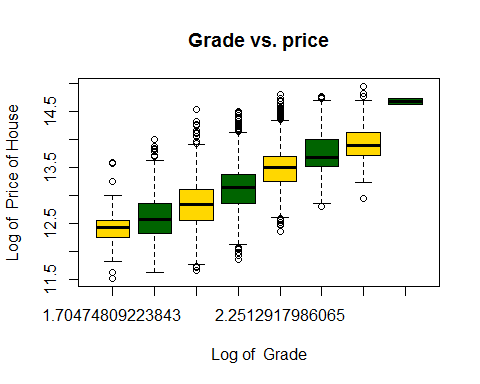
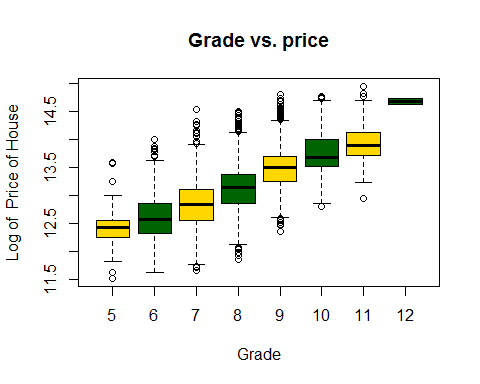
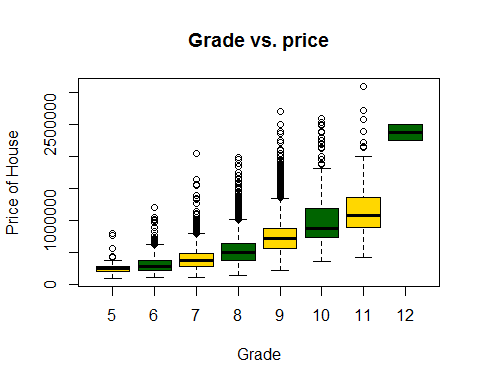
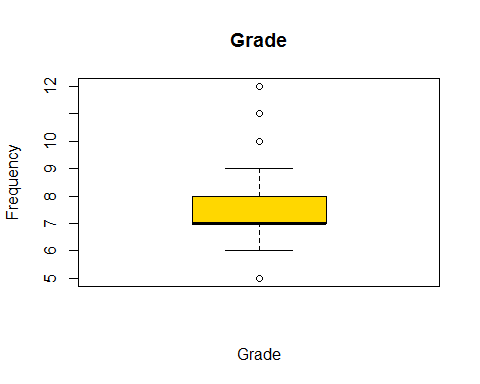
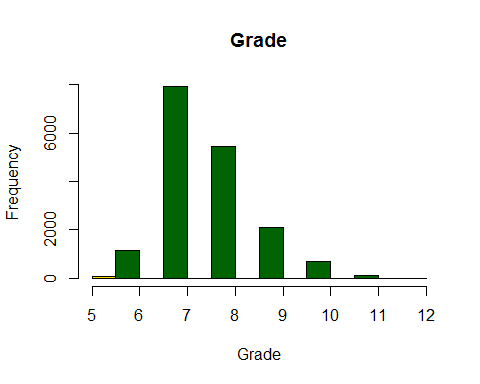
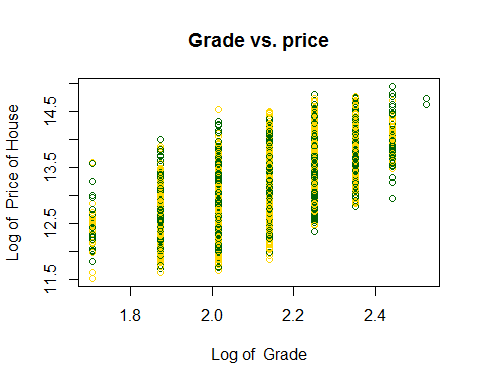
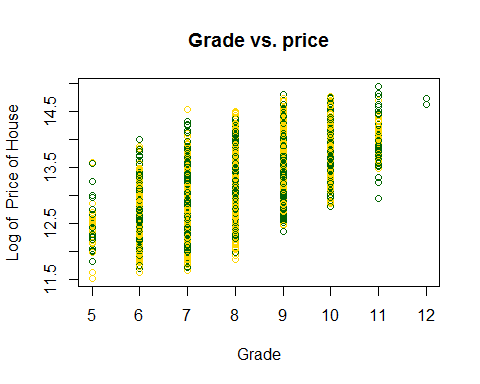
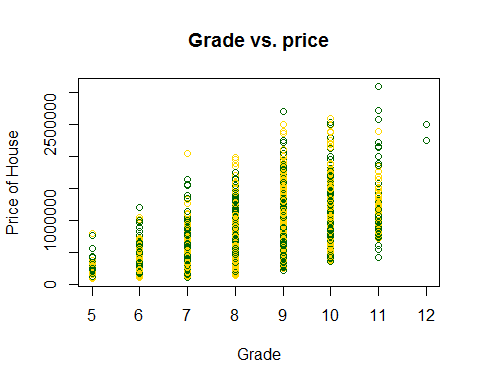


## [1] 0.06582398

#bucketByColumn(houseData,11)

## Grade Vs Price analysis

#bucketByColumn(houseData,12)  
analysis(houseData,12,c('Grade vs. price','Grade', 'Price of House'), 'Y', 'Y')

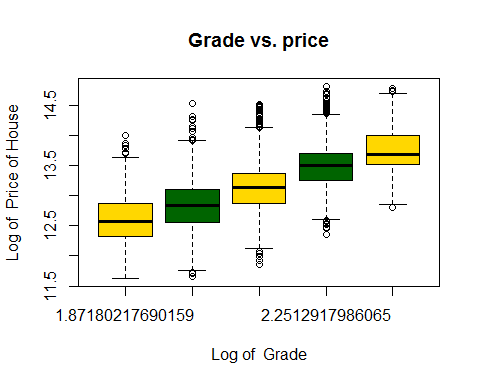
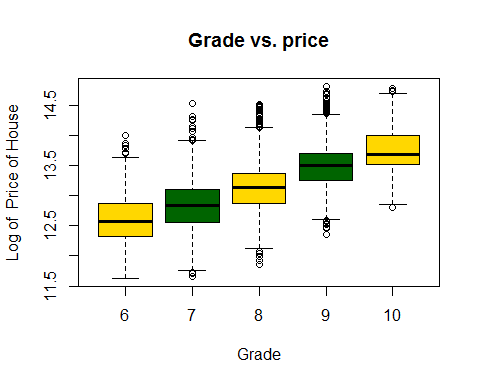
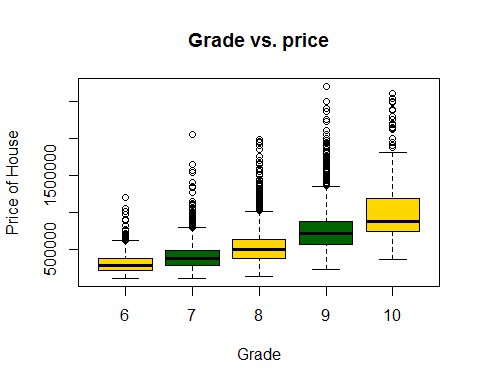
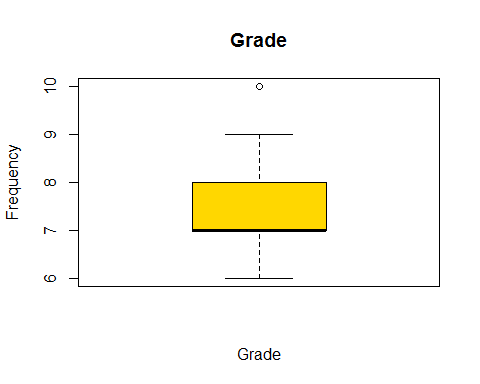
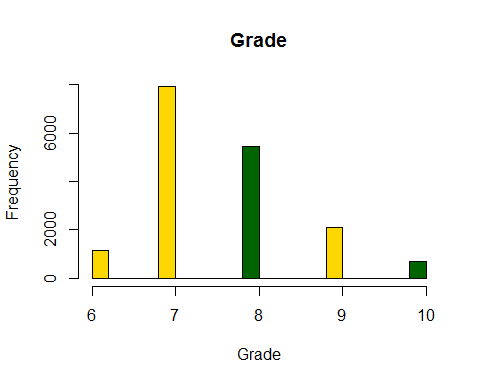
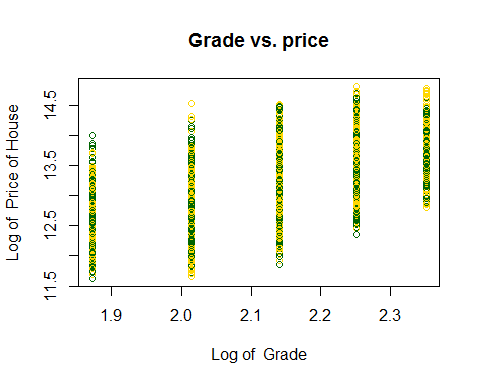
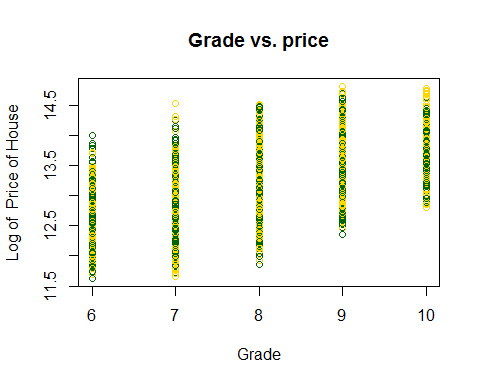
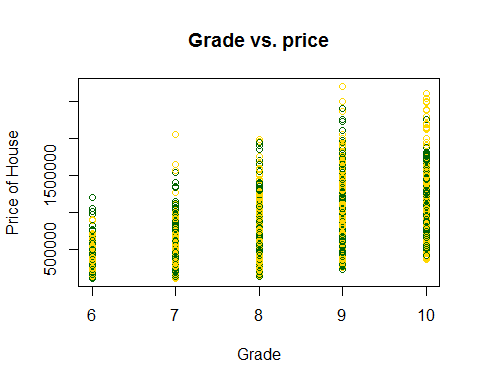


## [1] 0.6086724

cat ("Conclusion: Here we can se that as grade increases price also increases as well")

## Conclusion: Here we can se that as grade increases price also increases as well

#\*\*\*\*\*\*\*Removing the outliers  
#Most of the houses grades are between 6-10   
houseData<-subset(houseData,grade >= 6 & grade<=10)  
analysis(houseData,12,c('Grade vs. price','Grade', 'Price of House'), 'Y', 'Y')

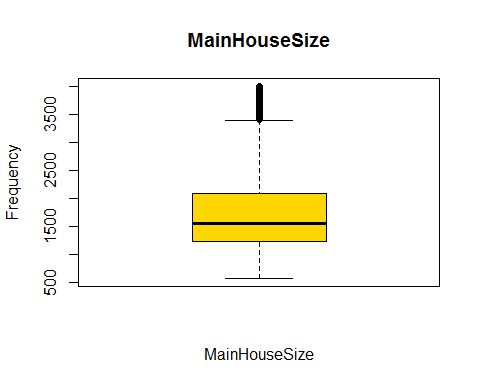
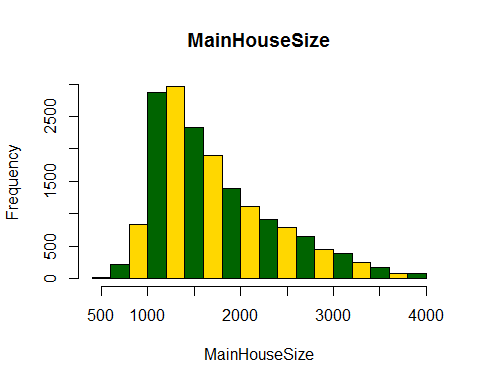
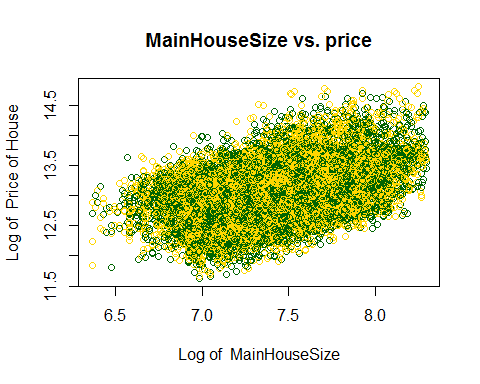
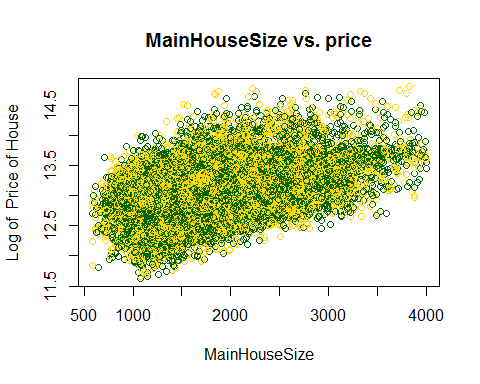
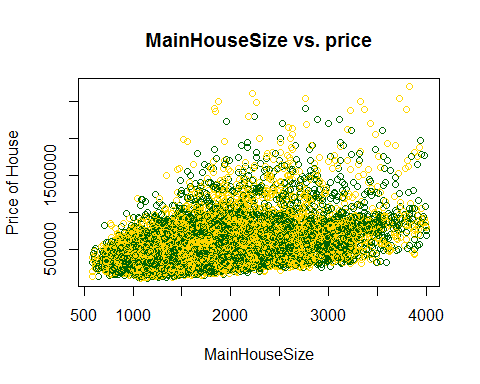


## [1] 0.5926375

#bucketByColumn(houseData,12)  
#grade is good without log

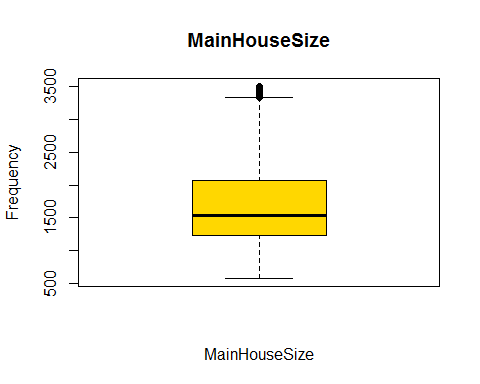
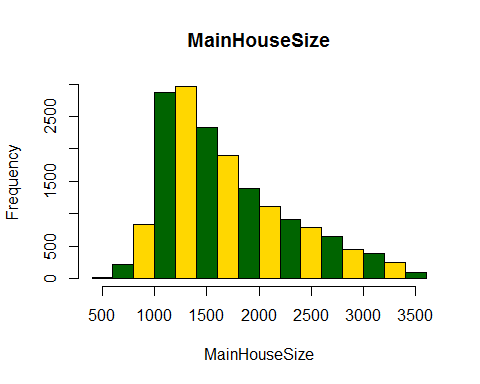
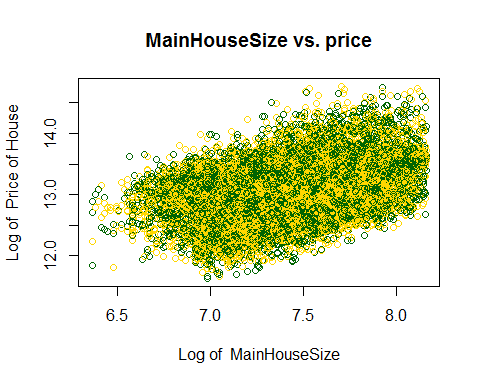
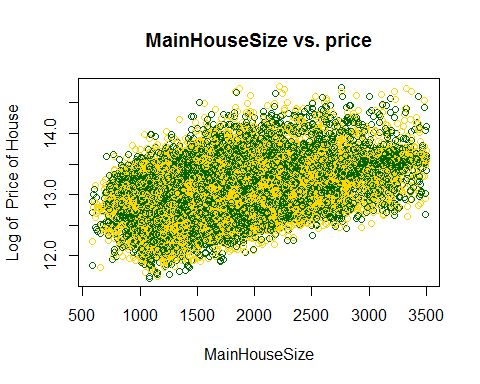
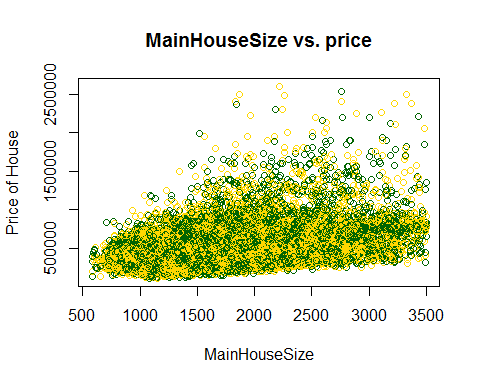
# SQFT\_ABOVE Vs Price analysis

#bucketByColumn(houseData,13)  
analysis(houseData,13,c('MainHouseSize vs. price','MainHouseSize', 'Price of House'), 'Y', 'N')



## [1] 0.4454655

#\*\*\*\*\*\*\*Removing the outliers  
houseData<-subset(houseData,sqft\_above >=500 & sqft\_above<=3500)  
analysis(houseData,13,c('MainHouseSize vs. price','MainHouseSize', 'Price of House'), 'Y', 'N')

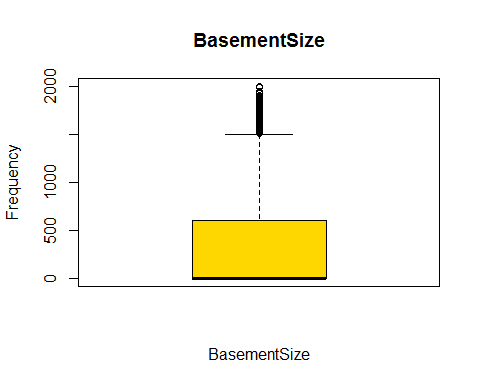
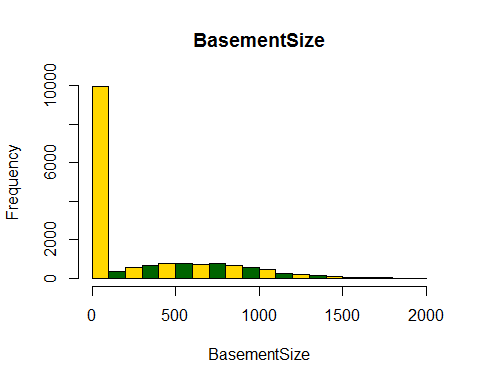
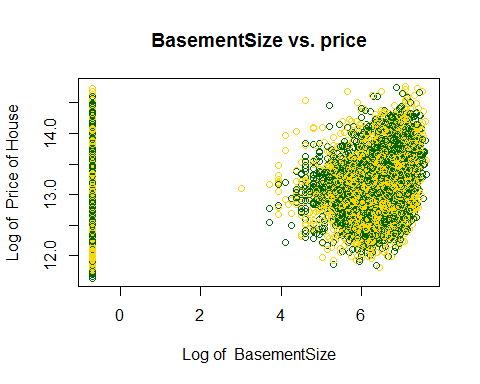
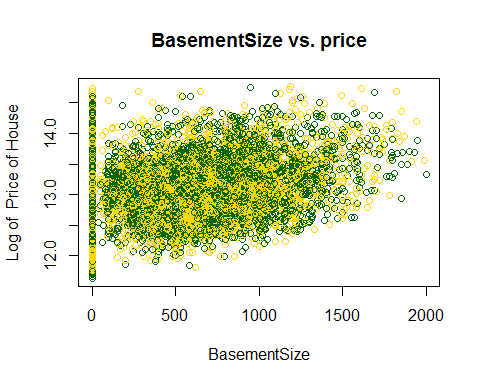
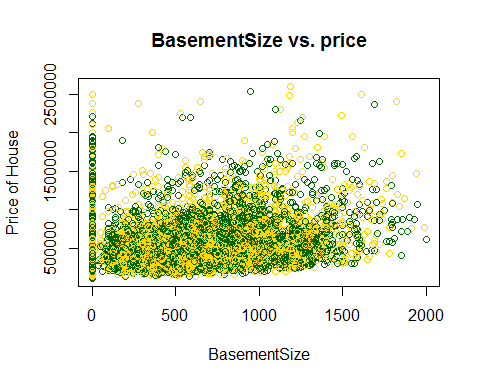


## [1] 0.4176138

#bucketByColumn(houseData,13)

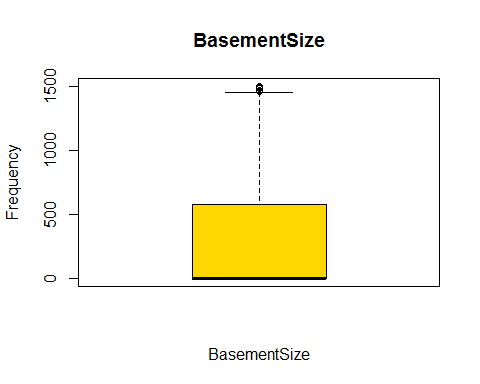
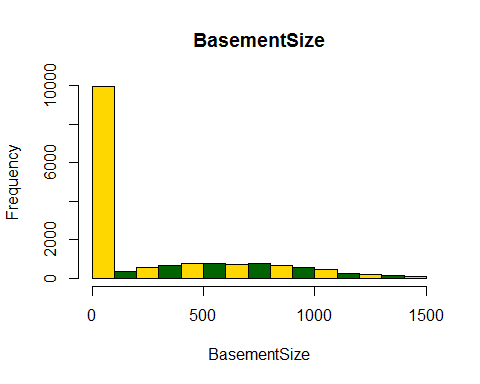
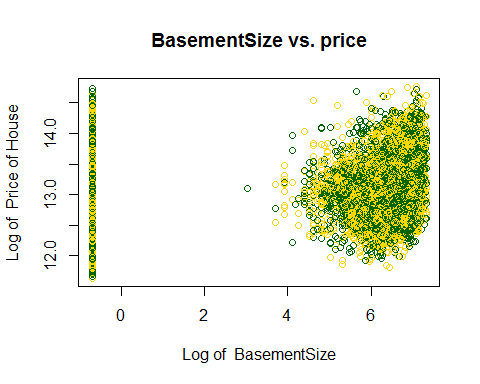
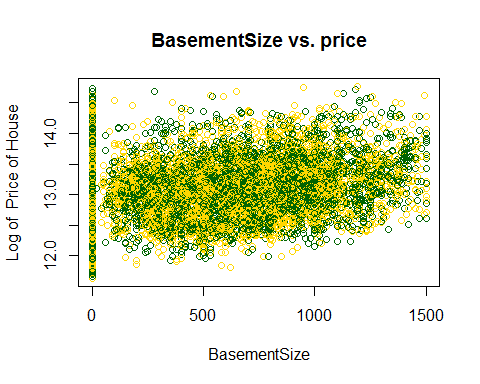
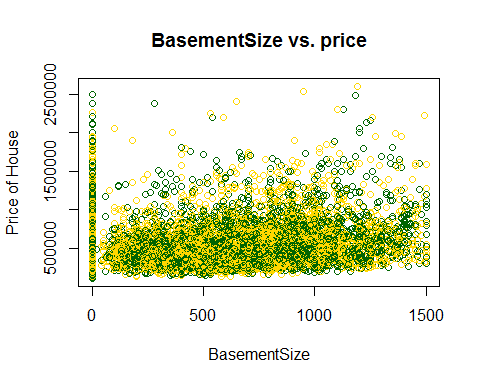
## SQFT\_BASEMENT Vs Price analysis

#bucketByColumn(houseData,14)  
analysis(houseData,14,c('BasementSize vs. price','BasementSize', 'Price of House'), 'Y', 'N')



## [1] 0.2686956

#\*\*\*\*\*\*\*Removing the outliers  
houseData<-subset(houseData,sqft\_basement >=0 & sqft\_basement<=1500)  
analysis(houseData,14,c('BasementSize vs. price','BasementSize', 'Price of House'), 'Y', 'N')

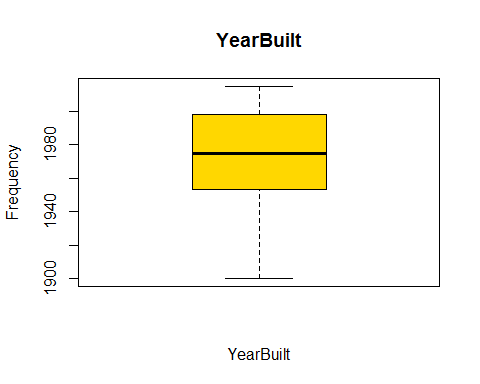
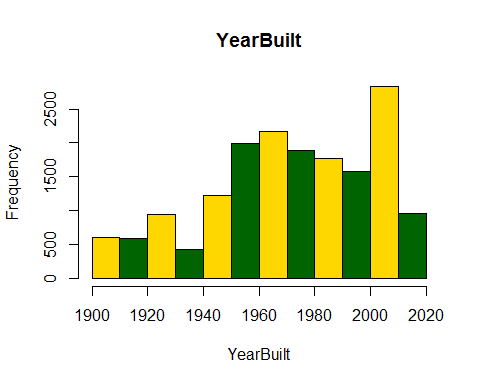
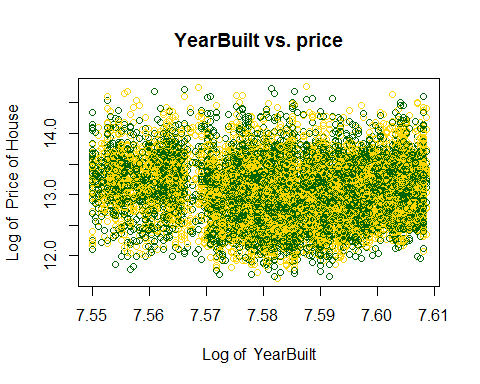
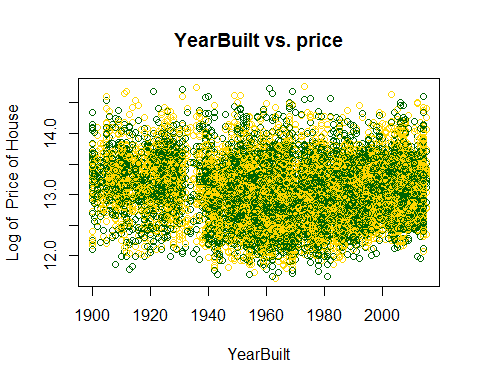
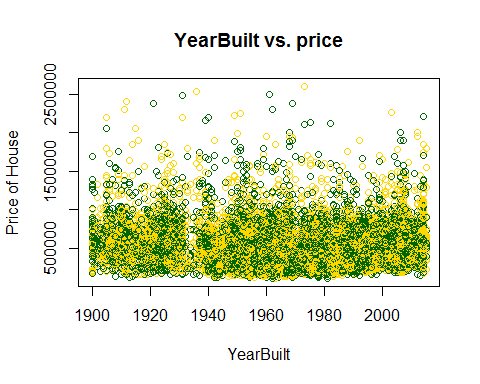


## [1] 0.2404668

#bucketByColumn(houseData,14)

# YR\_BUILT Vs Price analysis

#bucketByColumn(houseData,15)  
analysis(houseData,15,c('YearBuilt vs. price','YearBuilt', 'Price of House'), 'Y', 'N')

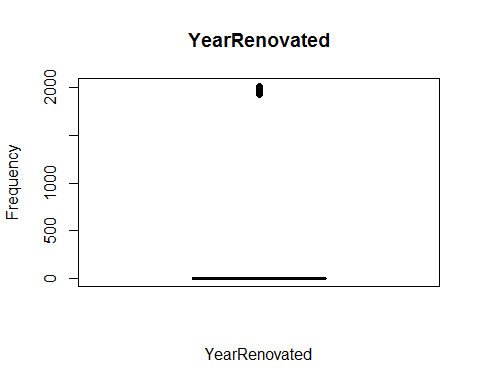
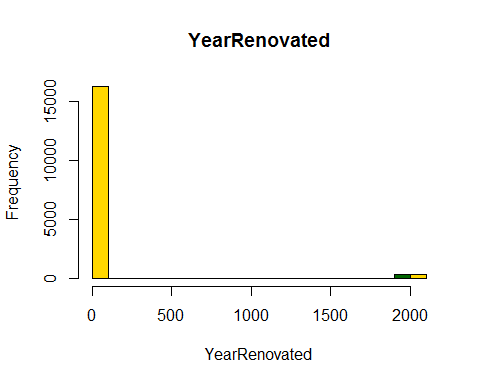
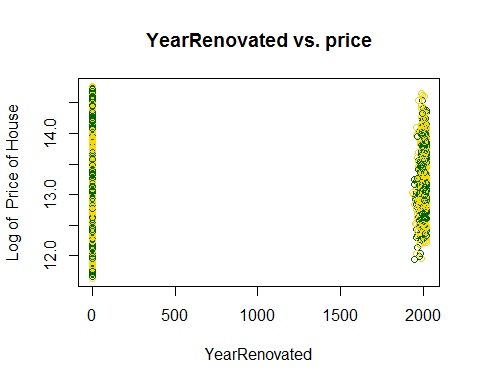
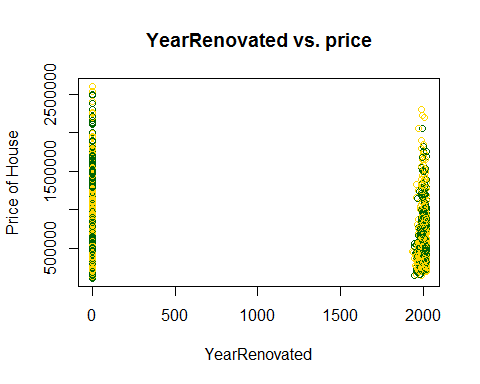


## [1] -0.09523834

#\*\*\*\*\*\*\*Removing the outliers  
#houseData<-subset(houseData,yr\_built>=1950& yr\_built<=2015)  
#analysis(houseData,15,c('YearBuilt vs. price','YearBuilt', 'Price of House'), 'Y', 'N')  
#bucketByColumn(houseData,15)

# YR\_RENOVATED Vs Price analysis

#bucketByColumn(houseData,16)  
analysis(houseData,16,c('YearRenovated vs. price','YearRenovated', 'Price of House'), 'N', 'N')

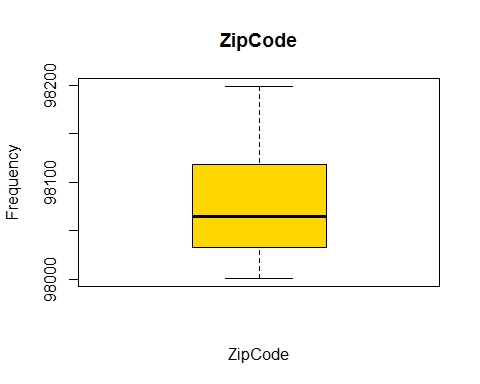
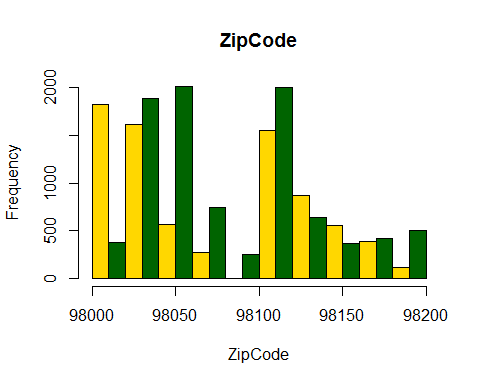
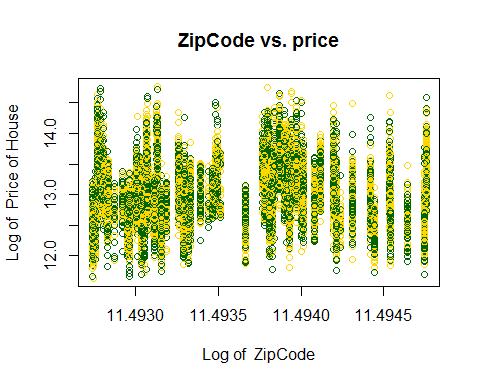
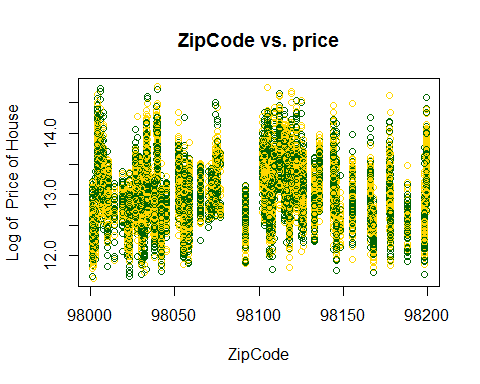
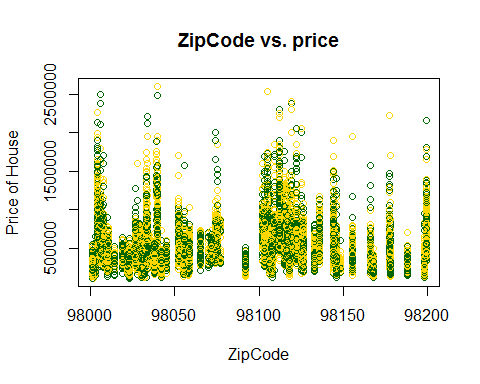


## [1] 0.1497455

#\*\*\*\*\*\*\*Removing the outliers  
#houseData<-subset(houseData,yr\_renovated>=1950& yr\_renovated<=2015)  
#analysis(houseData,16,c('YearRenovated vs. price','YearRenovated', 'Price of House'), 'N')  
#bucketByColumn(houseData,16)

# ZIPCODE Vs Price analysis

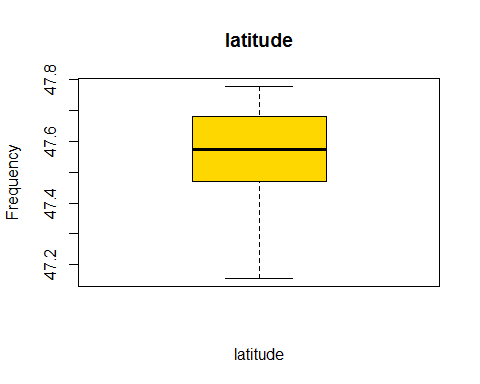
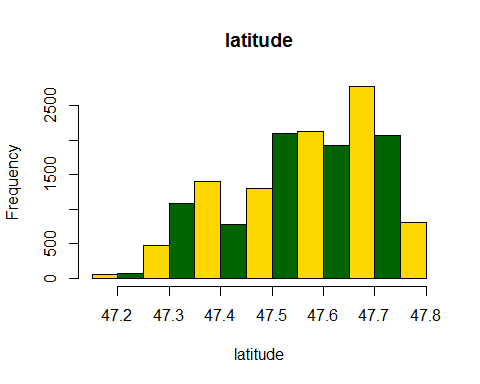
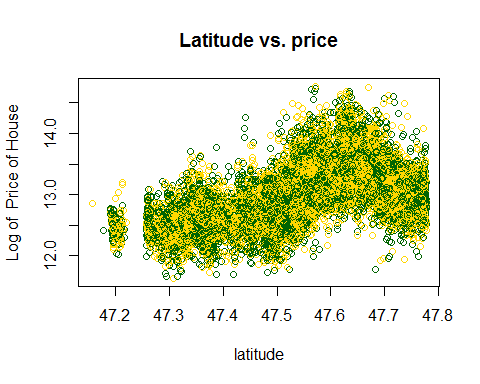
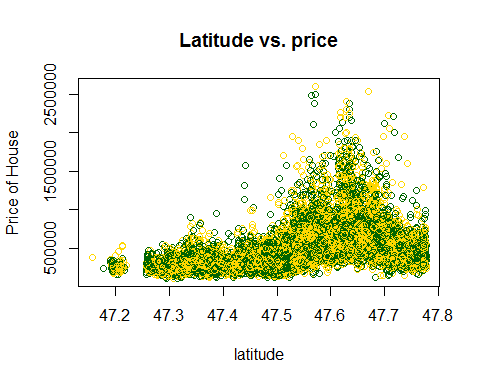
analysis(houseData,17,c('ZipCode vs. price','ZipCode', 'Price of House'), 'Y', 'N')



## [1] 0.04577519

## LAT Vs Price analysis

analysis(houseData,18,c('Latitude vs. price','latitude', 'Price of House'), 'N', 'N')

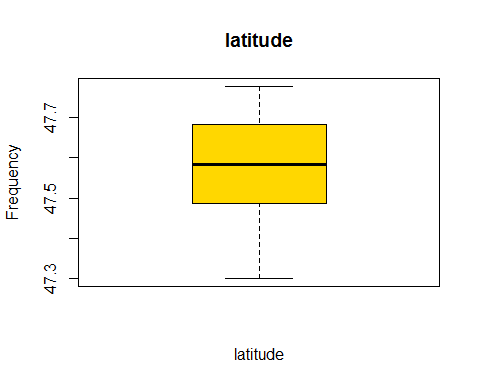
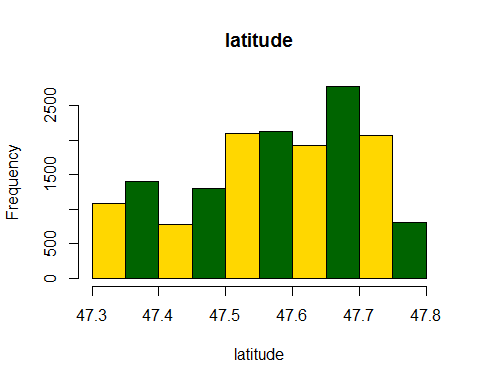
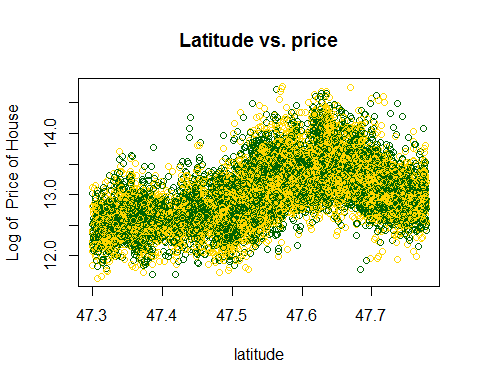
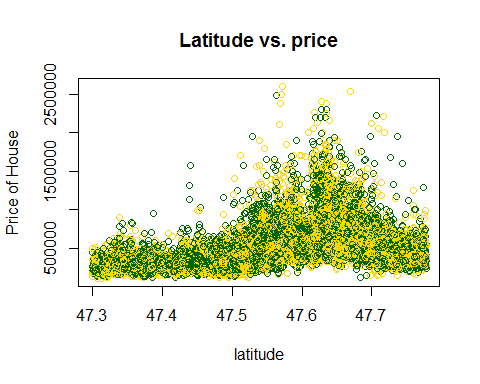


## [1] 0.4107748

cat ("Conclusion: Here we can see that LAThas normal dist relationship but not strong. So we should include LAT for price predication")

## Conclusion: Here we can see that LAThas normal dist relationship but not strong. So we should include LAT for price predication

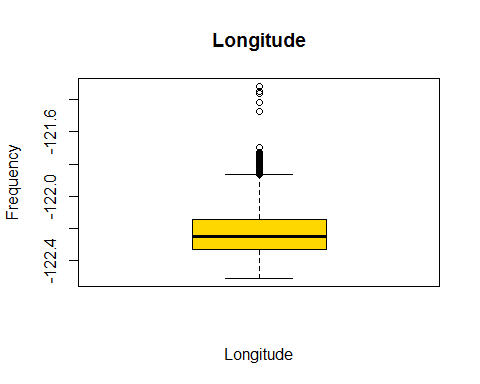
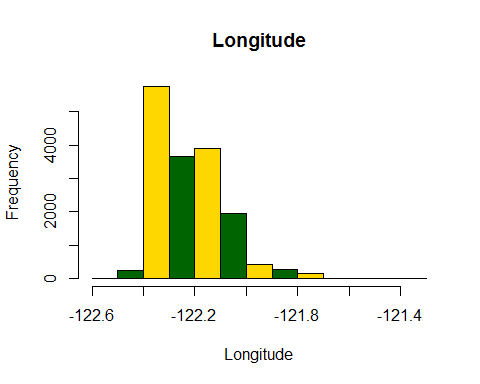
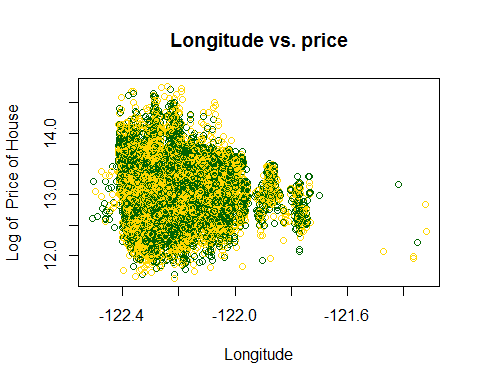
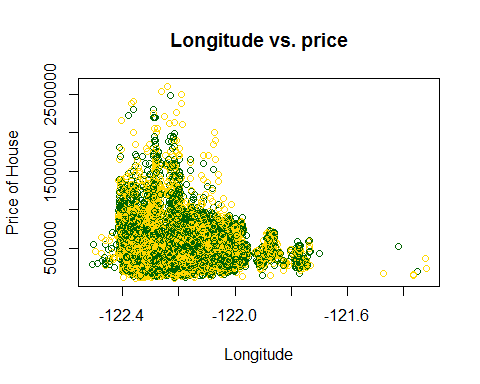
#\*\*\*\*\*\*\*Removing the outliers  
houseData<-subset(houseData,lat>=47.3)  
analysis(houseData,18,c('Latitude vs. price','latitude', 'Price of House'), 'N', 'N')



## [1] 0.3788783

## LONG Vs Price analysis

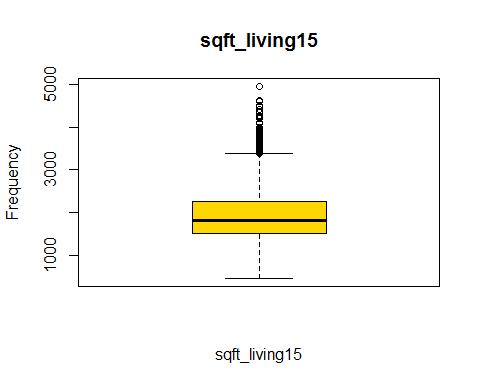
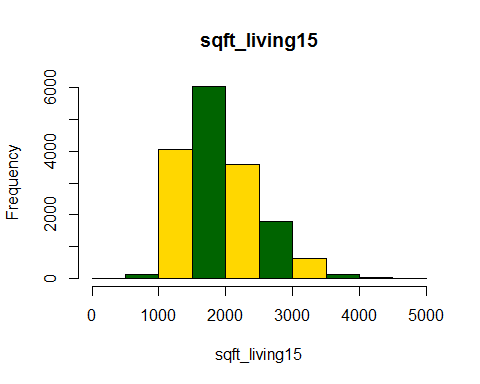
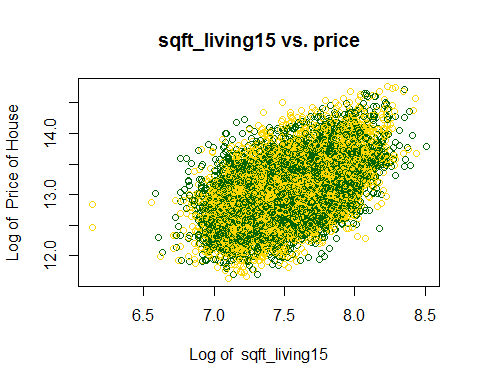
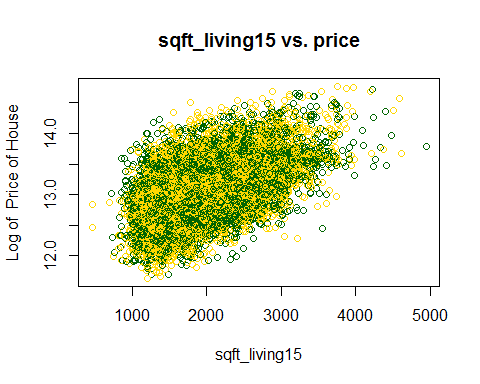
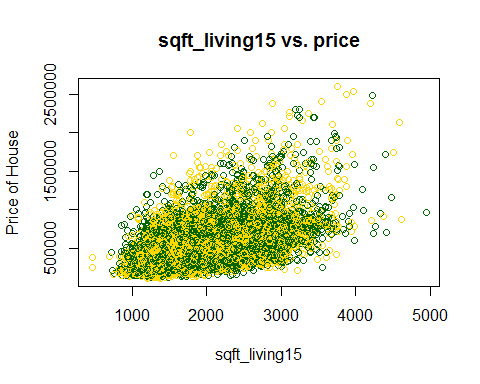
analysis(houseData,19,c('Longitude vs. price','Longitude', 'Price of House'), 'N', 'N')



## [1] -0.05515808

## SQFT\_LIVING15 Vs Price analysis

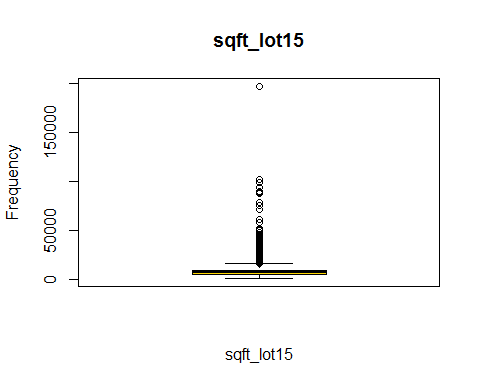
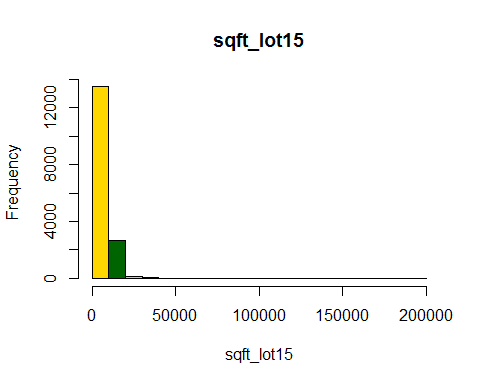
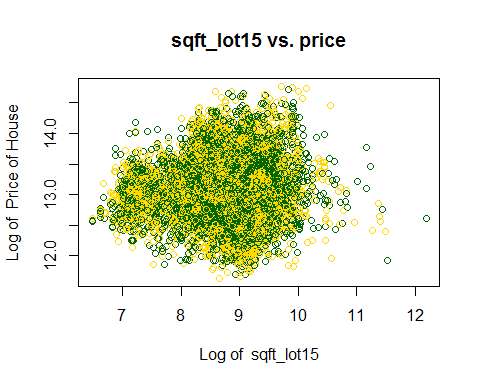
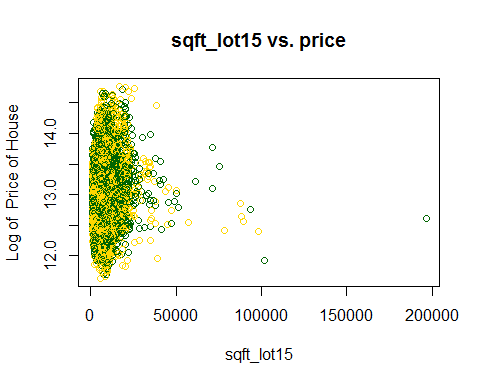
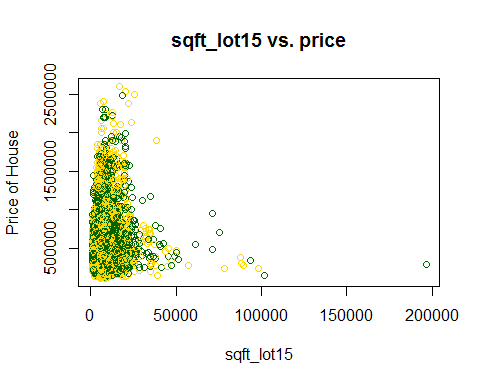
#bucketByColumn(houseData,20)  
analysis(houseData,20,c('sqft\_living15 vs. price','sqft\_living15', 'Price of House'), 'Y', 'N')



## [1] 0.5059352

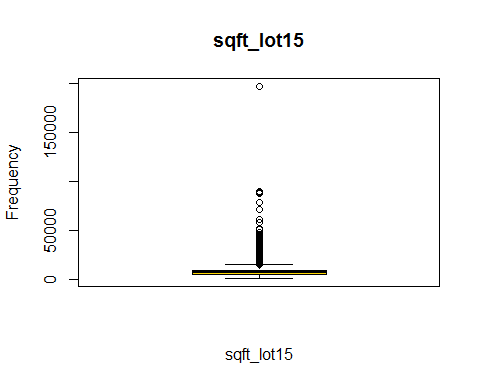
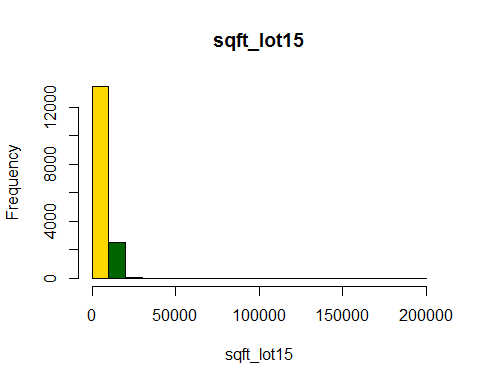
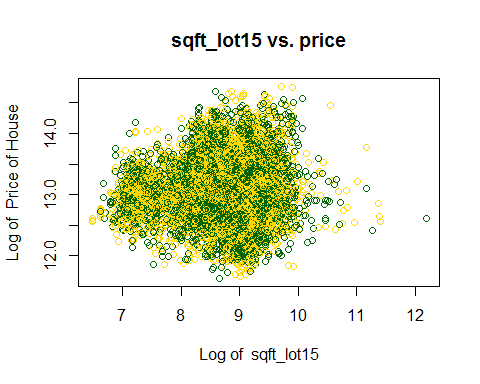
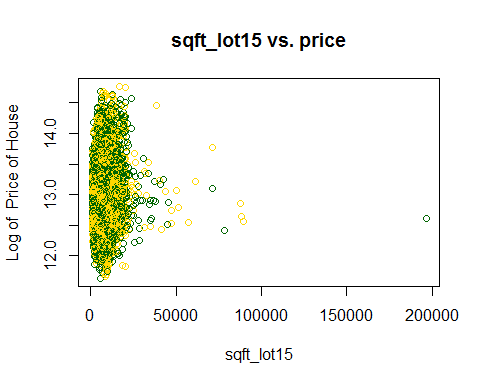
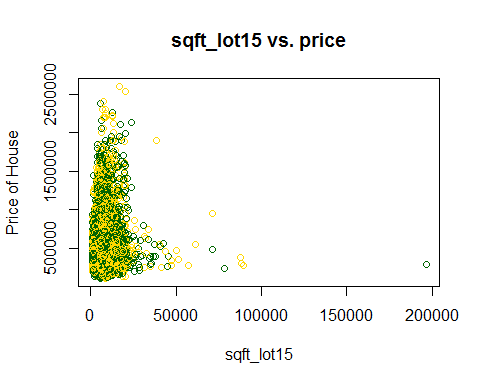
## SQFT\_LOT15 Vs Price analysis

#bucketByColumn(houseData,21)  
analysis(houseData,21,c('sqft\_lot15 vs. price','sqft\_lot15', 'Price of House'), 'Y', 'N')



## [1] 0.03039019

#\*\*\*\*\*\*\*Removing the outliers  
houseData<-subset(houseData,sqft\_lot15>=0 & sqft\_lot<=20000)  
analysis(houseData,21,c('sqft\_lot15 vs. price','sqft\_lot15', 'Price of House'), 'Y', 'N')



## [1] 0.01443069

#bucketByColumn(houseData,21)

## Correlation among all the variables

#houseData <- read.csv("file:///G:/Ryerson-BigData/capstone-R/CKME-136/data/kc\_house\_data.csv")  
#Remove the columns which does not hold any significance in predicing house price  
houseData$date <- NULL  
#houseData$id <- NULL  
#houseData$zipcode <- NULL  
cor(houseData)

## id price bedrooms bathrooms  
## id 1.0000000000 0.01143388 0.0002361529 0.03163163  
## price 0.0114338820 1.00000000 0.2091756651 0.36194911  
## bedrooms 0.0002361529 0.20917567 1.0000000000 0.39209060  
## bathrooms 0.0316316316 0.36194911 0.3920905987 1.00000000  
## sqft\_living 0.0367653294 0.56442385 0.5410387776 0.63702522  
## sqft\_lot -0.0455318043 0.01876577 0.1755961728 -0.06871884  
## floors 0.0212661816 0.20662923 0.0705733203 0.47816328  
## waterfront 0.0010194371 0.17456979 -0.0230699708 0.01753134  
## view 0.0240787568 0.34162165 0.0286568565 0.10107146  
## condition -0.0333233342 0.08514221 0.0328961173 -0.15089396  
## grade 0.0413583802 0.57797736 0.2098180353 0.54371647  
## sqft\_above 0.0432018940 0.43136334 0.4022870729 0.55180951  
## sqft\_basement -0.0081506170 0.23306924 0.2407207865 0.16269885  
## yr\_built 0.0361973063 -0.08917519 0.0557194457 0.49211680  
## yr\_renovated -0.0108237646 0.15279679 0.0158536971 0.04969098  
## zipcode -0.0358085658 0.01755183 -0.1129306298 -0.17401206  
## lat -0.0050349775 0.38499386 -0.0482258043 -0.02030436  
## long 0.0888037626 -0.05591129 0.0974013544 0.20522730  
## sqft\_living15 0.0441244099 0.49988474 0.3177246672 0.45383622  
## sqft\_lot15 -0.0499662192 0.01443069 0.1165777081 -0.05988074  
## sqft\_living sqft\_lot floors waterfront  
## id 0.036765329 -0.04553180 0.021266182 0.001019437  
## price 0.564423846 0.01876577 0.206629229 0.174569789  
## bedrooms 0.541038778 0.17559617 0.070573320 -0.023069971  
## bathrooms 0.637025219 -0.06871884 0.478163277 0.017531339  
## sqft\_living 1.000000000 0.19820432 0.270528403 0.041698559  
## sqft\_lot 0.198204325 1.00000000 -0.384503667 0.086919843  
## floors 0.270528403 -0.38450367 1.000000000 0.018659573  
## waterfront 0.041698559 0.08691984 0.018659573 1.000000000  
## view 0.200062577 0.08958453 -0.001229866 0.348892523  
## condition -0.053750363 0.14328773 -0.299656424 0.013488466  
## grade 0.629058762 0.03289696 0.428657101 0.039285103  
## sqft\_above 0.797370328 0.14354720 0.508622920 0.022631059  
## sqft\_basement 0.361775709 0.09409729 -0.354615993 0.031488303  
## yr\_built 0.229793174 -0.08248549 0.476688809 -0.026690358  
## yr\_renovated 0.057608432 0.00116957 0.006124387 0.081780181  
## zipcode -0.150371421 -0.24121431 -0.037895715 0.035364321  
## lat 0.008105012 -0.11580698 0.020066957 -0.015266662  
## long 0.194253942 0.23066503 0.105899291 -0.016162230  
## sqft\_living15 0.697868872 0.26243643 0.201674073 0.060720453  
## sqft\_lot15 0.142721794 0.73952278 -0.304480288 0.118857410  
## view condition grade sqft\_above  
## id 0.024078757 -0.03332333 0.04135838 0.04320189  
## price 0.341621655 0.08514221 0.57797736 0.43136334  
## bedrooms 0.028656857 0.03289612 0.20981804 0.40228707  
## bathrooms 0.101071460 -0.15089396 0.54371647 0.55180951  
## sqft\_living 0.200062577 -0.05375036 0.62905876 0.79737033  
## sqft\_lot 0.089584526 0.14328773 0.03289696 0.14354720  
## floors -0.001229866 -0.29965642 0.42865710 0.50862292  
## waterfront 0.348892523 0.01348847 0.03928510 0.02263106  
## view 1.000000000 0.04327161 0.18019550 0.07332838  
## condition 0.043271615 1.00000000 -0.17938320 -0.19041926  
## grade 0.180195500 -0.17938320 1.00000000 0.63891059  
## sqft\_above 0.073328382 -0.19041926 0.63891059 1.00000000  
## sqft\_basement 0.205531760 0.20850396 0.01545116 -0.27414378  
## yr\_built -0.100581799 -0.41152029 0.40106956 0.39365870  
## yr\_renovated 0.105176180 -0.06957926 0.01684730 0.01397774  
## zipcode 0.131494314 0.01435710 -0.15528743 -0.25245663  
## lat 0.018501159 0.01591340 0.07914349 -0.06721526  
## long -0.117206963 -0.12818534 0.15800734 0.34991253  
## sqft\_living15 0.236522082 -0.09507790 0.61776203 0.66135046  
## sqft\_lot15 0.094655883 0.12145254 0.03362976 0.10627697  
## sqft\_basement yr\_built yr\_renovated zipcode  
## id -0.008150617 0.03619731 -0.010823765 -0.03580857  
## price 0.233069237 -0.08917519 0.152796792 0.01755183  
## bedrooms 0.240720787 0.05571945 0.015853697 -0.11293063  
## bathrooms 0.162698848 0.49211680 0.049690975 -0.17401206  
## sqft\_living 0.361775709 0.22979317 0.057608432 -0.15037142  
## sqft\_lot 0.094097293 -0.08248549 0.001169570 -0.24121431  
## floors -0.354615993 0.47668881 0.006124387 -0.03789572  
## waterfront 0.031488303 -0.02669036 0.081780181 0.03536432  
## view 0.205531760 -0.10058180 0.105176180 0.13149431  
## condition 0.208503962 -0.41152029 -0.069579262 0.01435710  
## grade 0.015451165 0.40106956 0.016847300 -0.15528743  
## sqft\_above -0.274143782 0.39365870 0.013977744 -0.25245663  
## sqft\_basement 1.000000000 -0.24193394 0.070208939 0.15036865  
## yr\_built -0.241933942 1.00000000 -0.240636085 -0.33674162  
## yr\_renovated 0.070208939 -0.24063609 1.000000000 0.09140698  
## zipcode 0.150368655 -0.33674162 0.091406981 1.00000000  
## lat 0.116749091 -0.17857092 0.044945620 0.23426120  
## long -0.230988693 0.42995516 -0.096319359 -0.59345812  
## sqft\_living15 0.090438280 0.26659778 -0.019204514 -0.27296139  
## sqft\_lot15 0.063258054 -0.04944481 0.010334036 -0.19762198  
## lat long sqft\_living15 sqft\_lot15  
## id -0.005034977 0.08880376 0.04412441 -0.04996622  
## price 0.384993861 -0.05591129 0.49988474 0.01443069  
## bedrooms -0.048225804 0.09740135 0.31772467 0.11657771  
## bathrooms -0.020304356 0.20522730 0.45383622 -0.05988074  
## sqft\_living 0.008105012 0.19425394 0.69786887 0.14272179  
## sqft\_lot -0.115806977 0.23066503 0.26243643 0.73952278  
## floors 0.020066957 0.10589929 0.20167407 -0.30448029  
## waterfront -0.015266662 -0.01616223 0.06072045 0.11885741  
## view 0.018501159 -0.11720696 0.23652208 0.09465588  
## condition 0.015913405 -0.12818534 -0.09507790 0.12145254  
## grade 0.079143489 0.15800734 0.61776203 0.03362976  
## sqft\_above -0.067215262 0.34991253 0.66135046 0.10627697  
## sqft\_basement 0.116749091 -0.23098869 0.09043828 0.06325805  
## yr\_built -0.178570922 0.42995516 0.26659778 -0.04944481  
## yr\_renovated 0.044945620 -0.09631936 -0.01920451 0.01033404  
## zipcode 0.234261199 -0.59345812 -0.27296139 -0.19762198  
## lat 1.000000000 -0.16689129 -0.01114907 -0.10021621  
## long -0.166891288 1.00000000 0.32118284 0.19377056  
## sqft\_living15 -0.011149068 0.32118284 1.00000000 0.22930004  
## sqft\_lot15 -0.100216215 0.19377056 0.22930004 1.00000000

cat ("\nConclusion: sqft\_living , sqft\_above, grade, sqft\_living15, bathrooms have moderate to strong correlation with Price")

##   
## Conclusion: sqft\_living , sqft\_above, grade, sqft\_living15, bathrooms have moderate to strong correlation with Price

cat ("\nConclusion: bathrooms has moderate to strong correlation sqft\_living, floors, sqft\_above, grade, sqft\_living15, bedrooms")

##   
## Conclusion: bathrooms has moderate to strong correlation sqft\_living, floors, sqft\_above, grade, sqft\_living15, bedrooms

cat ("\nConclusion: sqft\_living has moderate to strong correlation sqft\_above, grade, sqft\_living15, bedrooms and all the above variables studied.")

##   
## Conclusion: sqft\_living has moderate to strong correlation sqft\_above, grade, sqft\_living15, bedrooms and all the above variables studied.

cat ("\nConclusion: grade has moderate to strong correlation sqft\_above, sqft\_living15 and all the above variables studied.")

##   
## Conclusion: grade has moderate to strong correlation sqft\_above, sqft\_living15 and all the above variables studied.

cat ("\nConclusion: sqft\_above has moderate to strong correlation sqft\_living15 and all the above variables studied.")

##   
## Conclusion: sqft\_above has moderate to strong correlation sqft\_living15 and all the above variables studied.

cat ("\nConclusion: sqft\_basement has moderate correlation with sqft\_living only.")

##   
## Conclusion: sqft\_basement has moderate correlation with sqft\_living only.

cat ("\nConclusion: yr\_built has moderate correlation with bathrooms, floors, grade, sqft\_above only.")

##   
## Conclusion: yr\_built has moderate correlation with bathrooms, floors, grade, sqft\_above only.

cat ("\nConclusion: sqft\_lot15 has strong correlation with sqft\_lot only.")

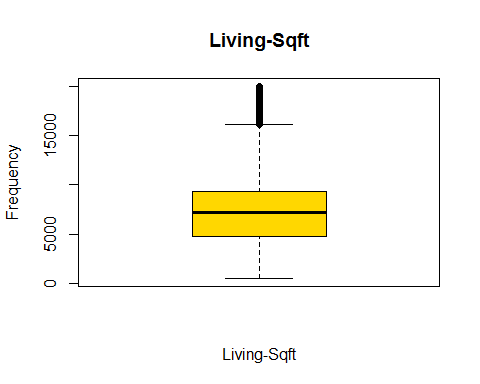
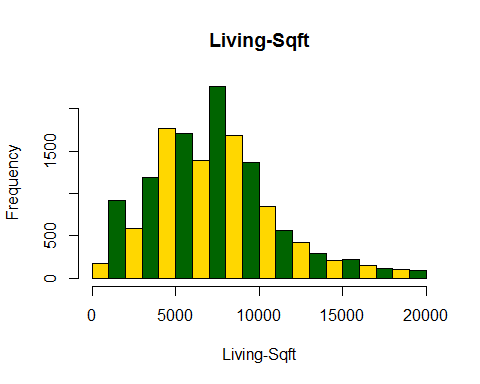
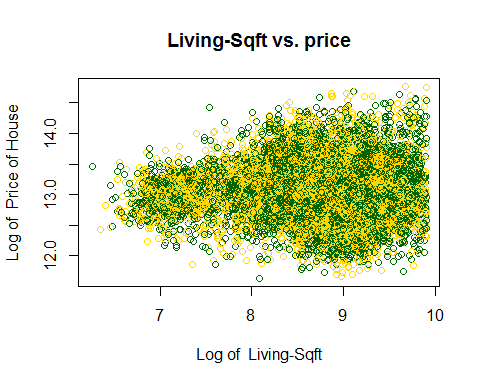
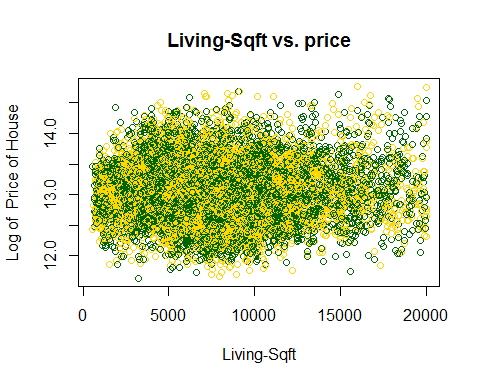
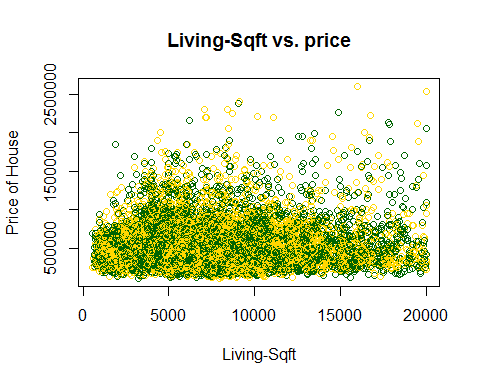
##   
## Conclusion: sqft\_lot15 has strong correlation with sqft\_lot only.

cat ("\nConclusion: waterfront, view, condition, zipcode, latitude, longitude, yr\_renovated has very weak with other attributes as well as with price.")

##   
## Conclusion: waterfront, view, condition, zipcode, latitude, longitude, yr\_renovated has very weak with other attributes as well as with price.

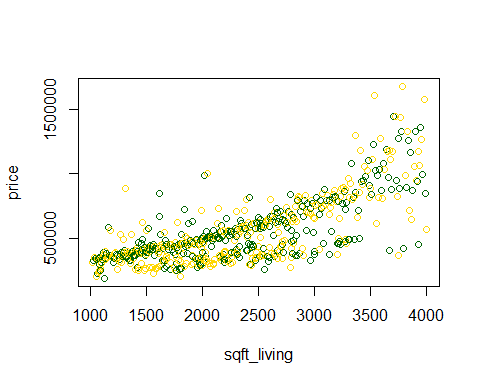
## Start with price & sqft\_living for Relationship

analysis(houseData,6,c('Living-Sqft vs. price','Living-Sqft', 'Price of House'), 'Y', 'N')



## [1] 0.01876577

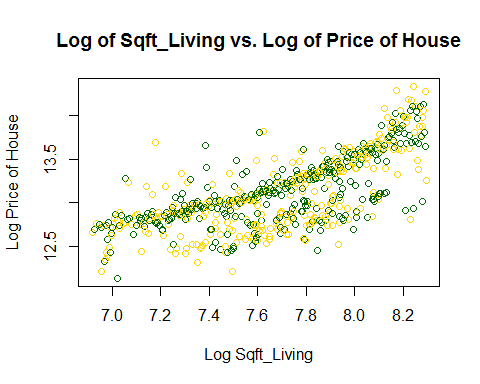
## Since above scatterplot is too crowded, I will plot aggregated vectors to verify the relationship between 2 (price, sqft\_living) variables.   
vec\_price\_sqftliving <-aggregate(price~sqft\_living, FUN=mean, data=houseData)  
plot(vec\_price\_sqftliving, col=(c("gold","darkgreen")))



scatterplot1<-recordPlot()  
  
cat ("\nConclusion: Plot does not show that price and sqft\_living are linearly related. It looks like an exponential relationship.")

##   
## Conclusion: Plot does not show that price and sqft\_living are linearly related. It looks like an exponential relationship.

plot(log(vec\_price\_sqftliving$sqft\_living),log(vec\_price\_sqftliving$price), main="Log of Sqft\_Living vs. Log of Price of House", xlab="Log Sqft\_Living", ylab="Log Price of House", col=(c("gold","darkgreen")))



scatterplot2<-recordPlot()  
  
cat ("\nConclusion: Relationship between variables in plot 1 seems to be exponential and in plot 2 it seems to be linear.")

##   
## Conclusion: Relationship between variables in plot 1 seems to be exponential and in plot 2 it seems to be linear.

## My fisrt 5 variables are: sqft\_living, bathrooms, grade, view, sqft\_living15, sqft\_above.

## Each of box plots shows that above variables might be directly related in predicting house prices.

## To support my finding, I also computed correlation between prices and variables, and my top 5 picks are supported with correlation coefficients as well [see below]

# corr between price vs sqft\_living: 0.70203505

# corr between price vs bathrooms: 0.52513751

# corr between price vs bedrooms: 0.308349598

# corr between price vs floors: 0.256793888

# corr between price vs waterfront: 0.266369434

# corr between price vs view: 0.397293488

# corr between price vs grade: 0.66743426

# corr between price vs sqft\_above: 0.6055672984

# corr between price vs lat: 0.3070034800

# corr between price vs sqft\_living15: 0.58537890

## Prepare the data for the Model

## 1: seed is being set, so that distribution is same at each run

## 2: put 70% to train-set

## 3: put 30% to test-set

set.seed(1)  
rn\_train <- sample(nrow(houseData),floor(nrow(houseData)\*0.70))  
train <- houseData[rn\_train,colnames(houseData)]  
test <- houseData[-rn\_train,colnames(houseData)]

## Selection Method: START

## We have suggested 12 variables. I will calculate SSE and see which one gives me a smaller SSE and I pick that variable.

## From above correlation, variables selected are: bedrooms, bathrooms, log(sqft living), log(sqft lot), floors, waterfront, view, grade, yr built, lat

## 

## ###### stepByStep Analysis - Step -01

## ###### One variable predication - START

## 

## Creating Models using 1 variables fpr each of the variables with price, so total we have 12 Models.   
SSEVals <- list(c(), c())  
SSEVals = findSSEByColName('bedrooms', SSEVals)  
SSEVals = findSSEByColName('bathrooms', SSEVals)  
SSEVals = findSSEByColName('log(sqft\_living)', SSEVals)  
SSEVals = findSSEByColName('log(sqft\_lot)', SSEVals)  
SSEVals = findSSEByColName('floors', SSEVals)  
SSEVals = findSSEByColName('waterfront', SSEVals)  
SSEVals = findSSEByColName('view', SSEVals)  
SSEVals = findSSEByColName('grade', SSEVals)  
SSEVals = findSSEByColName('yr\_built', SSEVals)  
SSEVals = findSSEByColName('lat', SSEVals)  
SSEArr = SSEVals[[2]];  
which(SSEArr==min(SSEArr))

## [1] 8

SSEVals[[1]][which(SSEArr==min(SSEArr))]

## [1] "grade"

cat("\nSTEP -01 : Conculsion: SSE is minimum for variable 'grade' so it is the best predictor, when we use single variable")

##   
## STEP -01 : Conculsion: SSE is minimum for variable 'grade' so it is the best predictor, when we use single variable

## 

## ###### stepByStep Analysis - Step -01

## ###### One variable predication - End

## 

## 

## ###### stepByStep Analysis - Step -02

## ###### Add more attributes to model - START

## 

# we will also use R-squared to measure accuracy of model with addition of more variable

# R-squared is a statistical measure of how close the data are to the fitted regression line. It is also known as the coefficient of determination, or the coefficient of multiple determination for multiple regression. 0% indicates that the model explains none of the variability of the response data around its mean

# As in last phase, grade is bet to predict the price with single variable  
# let us name is model01 & compute r\_square for the model  
# Linear Model  
model01 <- lm(data=train,log(price)~grade)  
summary(model01)

##   
## Call:  
## lm(formula = log(price) ~ grade, data = train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.30048 -0.25702 0.00205 0.24136 1.67255   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 10.837314 0.028973 374.05 <2e-16 \*\*\*  
## grade 0.289070 0.003795 76.16 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3612 on 11256 degrees of freedom  
## Multiple R-squared: 0.3401, Adjusted R-squared: 0.34   
## F-statistic: 5801 on 1 and 11256 DF, p-value: < 2.2e-16

# R-Square of Model  
r\_squared\_model01<-summary(model01)$r.squared  
# RMSE of Model  
predic\_model01<-exp(predict(model01,interval='prediction',newdata=test))   
RMSE\_model01=sqrt(sum((predic\_model01 - test$price)^2)/nrow(test))

# Now let us add more variable which has greater impact on the price prediction.

# Let us call it model02.

# Approach of attribute selection:

# I have selecetd log(sqft\_living), bedrooms, bathrooms, grade, waterfront

# here I have tried to make mix-match of the variables which have strong/moderate correlation with price, but variable do not strong co-orelation between then

# Linear Model  
model02<-lm(log(price)~log(sqft\_living)+bedrooms+bathrooms+grade+waterfront,data=train)  
summary(model02)

##   
## Call:  
## lm(formula = log(price) ~ log(sqft\_living) + bedrooms + bathrooms +   
## grade + waterfront, data = train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.2946 -0.2457 0.0055 0.2330 1.3062   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 7.599956 0.098861 76.875 < 2e-16 \*\*\*  
## log(sqft\_living) 0.555118 0.016358 33.936 < 2e-16 \*\*\*  
## bedrooms -0.033813 0.004836 -6.992 2.86e-12 \*\*\*  
## bathrooms -0.043236 0.006751 -6.404 1.57e-10 \*\*\*  
## grade 0.190612 0.004743 40.185 < 2e-16 \*\*\*  
## waterfront 0.632131 0.048576 13.013 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3385 on 11252 degrees of freedom  
## Multiple R-squared: 0.4206, Adjusted R-squared: 0.4203   
## F-statistic: 1633 on 5 and 11252 DF, p-value: < 2.2e-16

# R-Square of Model  
r\_squared\_model02<-summary(model02)$r.squared  
##compute RMSE for Model-02  
predic\_model02<-exp(predict(model02,interval='prediction',newdata=test))   
RMSE\_model02=sqrt(sum((predic\_model02 - test$price)^2)/nrow(test))   
  
cat("\nR-Squared for Model-02 is ",100\*(r\_squared\_model02/r\_squared\_model01-1),"% better than Model-01.")

##   
## R-Squared for Model-02 is 23.66765 % better than Model-01.

cat("\nSTEP -02 : Conculsion: \n Model-02 will predict the price better tha Model-01")

##   
## STEP -02 : Conculsion:   
## Model-02 will predict the price better tha Model-01

predic\_model02<-exp(predict(model02,newdata=test))

## 

## ###### stepByStep Analysis - Step -02

## ###### Add more attributes to model - END

## 

## 

## ###### stepByStep Analysis - Step -03

## ## new attributes using Residual approach

## ########## START

## 

## If we plot residual vs. a variable (that is not used in the prediction) and if we see any recognizable patterns,

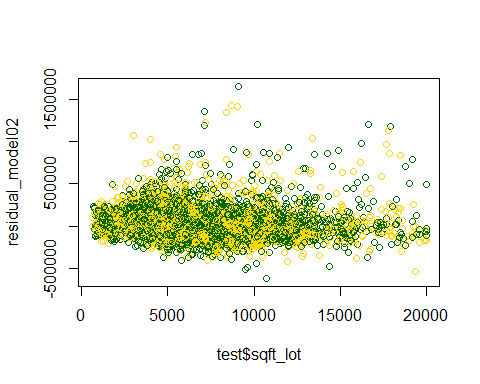
## then it indicates that some of the variation in residual is due to non-used variable

## therefore we should include it in our model to reduce the residual errors.

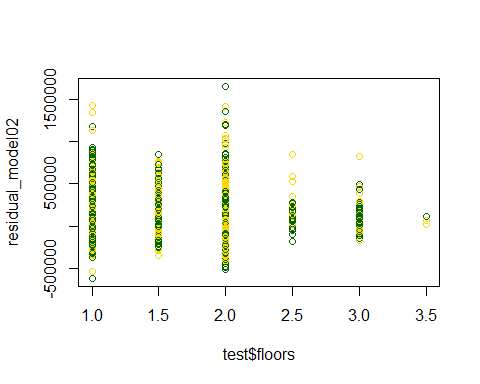
## To calculate residuals,

## we simply need to substract predic\_model02 from the actual price.

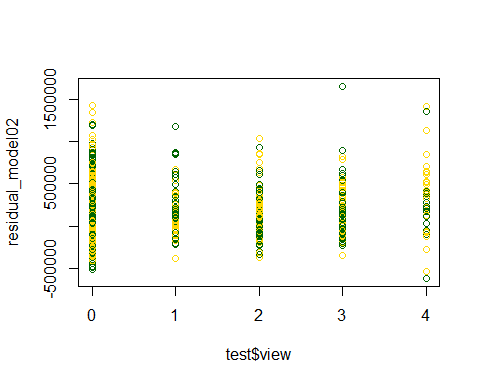
##calculate residual for Model-02  
residual\_model02=test$price - predic\_model02  
## Residual vs. sqft\_lot  
plot(test$sqft\_lot,residual\_model02, col=(c("gold","darkgreen")))



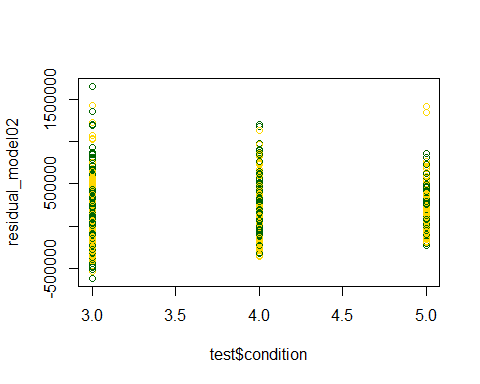
## Residual vs. floors  
plot(test$floors,residual\_model02, col=(c("gold","darkgreen")))



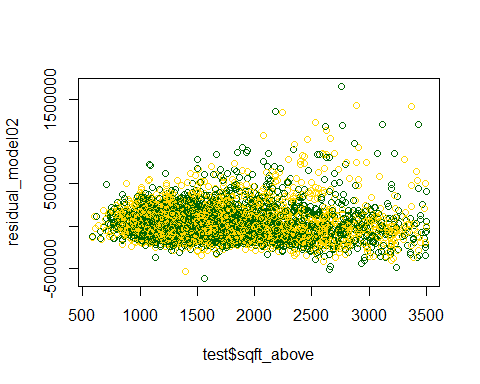
## Residual vs. view  
plot(test$view,residual\_model02, col=(c("gold","darkgreen")))



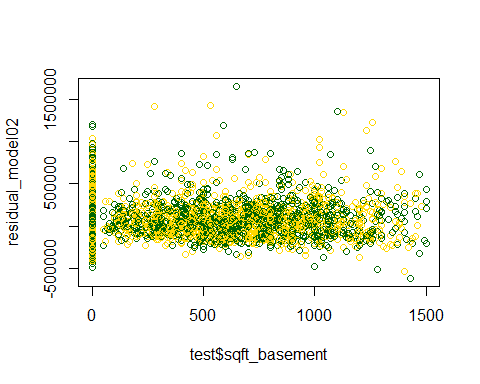
## Residual vs. condition  
plot(test$condition,residual\_model02, col=(c("gold","darkgreen")))



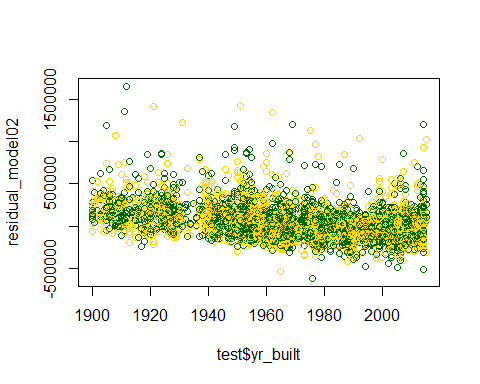
## Residual vs. sqft\_above  
plot(test$sqft\_above,residual\_model02, col=(c("gold","darkgreen")))



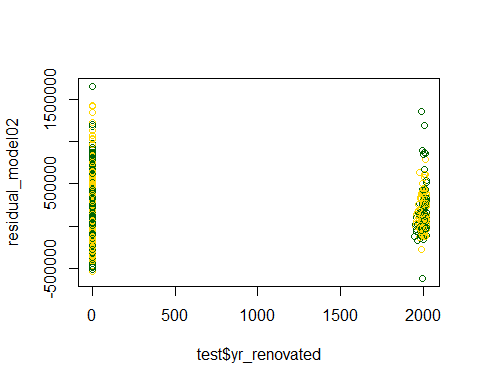
## Residual vs. sqft\_basement  
plot(test$sqft\_basement,residual\_model02, col=(c("gold","darkgreen")))



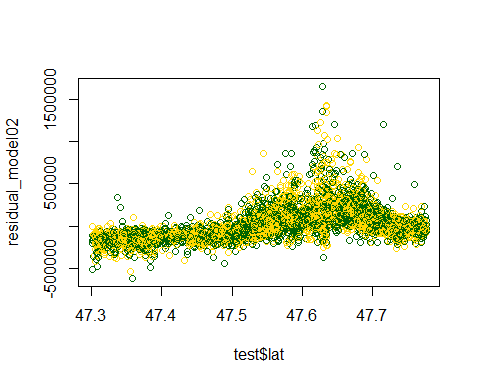
## Residual vs. yr\_built  
plot(test$yr\_built,residual\_model02, col=(c("gold","darkgreen")))



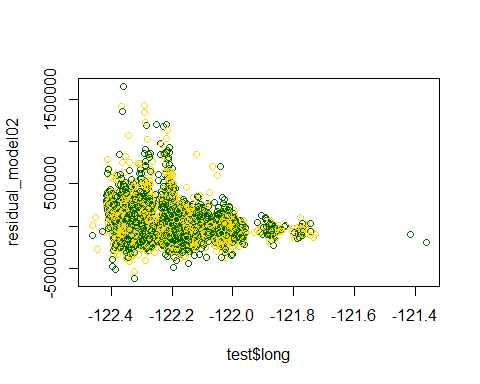
## Residual vs. yr\_renovated  
plot(test$yr\_renovated,residual\_model02, col=(c("gold","darkgreen")))



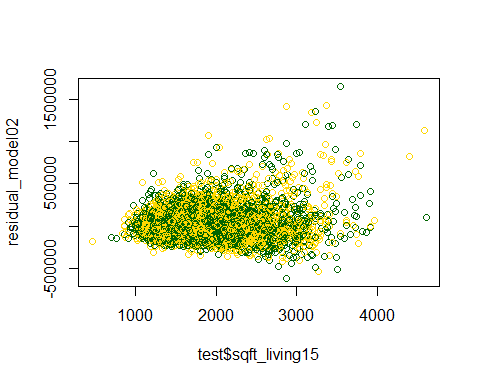
## Residual vs. zipcode  
#plot(test$zipcode,residual\_model02, col=(c("gold","darkgreen")))   
  
## Residual vs. lat  
plot(test$lat,residual\_model02, col=(c("gold","darkgreen")))



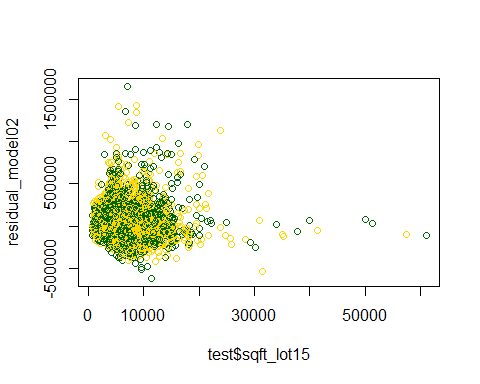
## Residual vs. long  
plot(test$long,residual\_model02, col=(c("gold","darkgreen")))



## Residual vs. sqft\_living15  
plot(test$sqft\_living15,residual\_model02, col=(c("gold","darkgreen")))



## Residual vs. sqft\_lot15  
plot(test$sqft\_lot15,residual\_model02, col=(c("gold","darkgreen")))



cat("\nSTEP -03 : Conculsion: After analzing scattered plot, we found that yr\_built & lat are good candidate to predict the price of house")

##   
## STEP -03 : Conculsion: After analzing scattered plot, we found that yr\_built & lat are good candidate to predict the price of house

## 

## ###### stepByStep Analysis - Step -03

## ## new attributes using Residual approach

## ########## End

## 

## 

## ###### stepByStep Analysis - Step -04

## ## Residual approach and get new model

## ########## STRAT

## 

## Below two columns is being added

## 1: yr\_built

## 2:lat

#Now create model03, which will include yr\_built & lat and we will try to find model03 vs model02  
model03<-lm(log(price)~log(sqft\_living)+bedrooms+bathrooms+grade+waterfront+yr\_built+lat,data=train)  
summary(model03)

##   
## Call:  
## lm(formula = log(price) ~ log(sqft\_living) + bedrooms + bathrooms +   
## grade + waterfront + yr\_built + lat, data = train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.19597 -0.15767 0.00181 0.15816 1.27443   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -5.016e+01 9.654e-01 -51.959 <2e-16 \*\*\*  
## log(sqft\_living) 4.568e-01 1.226e-02 37.270 <2e-16 \*\*\*  
## bedrooms -3.299e-02 3.576e-03 -9.225 <2e-16 \*\*\*  
## bathrooms 8.168e-02 5.503e-03 14.843 <2e-16 \*\*\*  
## grade 2.042e-01 3.641e-03 56.077 <2e-16 \*\*\*  
## waterfront 5.875e-01 3.587e-02 16.381 <2e-16 \*\*\*  
## yr\_built -4.331e-03 9.848e-05 -43.981 <2e-16 \*\*\*  
## lat 1.402e+00 1.887e-02 74.267 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.2494 on 11250 degrees of freedom  
## Multiple R-squared: 0.6856, Adjusted R-squared: 0.6854   
## F-statistic: 3505 on 7 and 11250 DF, p-value: < 2.2e-16

r\_squared\_model03<-summary(model03)$r.squared  
  
cat("\nR-Squared for Model-03 is ",100\*(r\_squared\_model03/r\_squared\_model02-1),"% better than Model-02.\nR-squared for Model-03 and Model-02 are:", r\_squared\_model03,"and", r\_squared\_model02, "respectively.So here Model-03 wins over other previous model.")

##   
## R-Squared for Model-03 is 63.02309 % better than Model-02.  
## R-squared for Model-03 and Model-02 are: 0.6856254 and 0.4205695 respectively.So here Model-03 wins over other previous model.

## RMSE for Model-03:  
predic\_model03<-exp(predict(model03,interval='prediction',newdata=test))  
RMSE\_model03=sqrt(sum((predic\_model03 - test$price)^2)/nrow(test))  
  
cat("\nRMSE for model02:",RMSE\_model02,"\nRMSE for model03:",RMSE\_model03)

##   
## RMSE for model02: 614498.8   
## RMSE for model03: 465951.2

cat("\nConclusion: RMSE for Model-02 is ",round(100\*(RMSE\_model02/RMSE\_model03-1),2),"% more than Model-03. So Model-03 predicts the prices better.")

##   
## Conclusion: RMSE for Model-02 is 31.88 % more than Model-03. So Model-03 predicts the prices better.

## 

## ###### stepByStep Analysis - Step -04

## ## Residual approach and get new model

## ########## END

## 

## 

## ###### stepByStep Analysis - Step -05

## ## Tip/Formula: what you will look for

## ######### when you buy a house

## ## enrich model with domain knowlege

## ################ START

## 

# log(price) seems to have good correlation with exp(bathrooms).

# log(price) seems to have good correlation with log(bedrooms), add 0.05 as it has 0 values.

# log(price) seems to have good correlation with log(lat). to reduce the noice and we will use lat-min(lat)+0.5, so that relative relation can be found. Similar approach has been taken for long

# log(price) seems to have good correlation with log(yr\_renovated). to get more relevent data we wil try to compute the age of the building

# bedroom & bathroom can also play a big role when price of an house is being calculate, so i have use bedrooms*bathrooms as one of the variable to compute linear model #grade & condition can also play a big role when price of an house is being calculate, so i have use log(grade)*exp(condition) as one of the variable to compute linear model

model05<-lm(log(price)~grade+log(sqft\_living)+log(bedrooms+0.5)+exp(bathrooms)+waterfront+log(abs(lat-min(lat))+0.5)+log(abs(long-min(long))+0.05)+(zipcode\*lat)+log(view+0.5)+condition+log(sqft\_above+0.05)+log(sqft\_basement+0.05)+log(sqft\_lot15)+log(2015-yr\_renovated+1)+(bedrooms\*bathrooms)+(log(grade)\*exp(condition))+(bedrooms\*log(sqft\_living))+(view\*bedrooms),data=train)  
summary(model05)

##   
## Call:  
## lm(formula = log(price) ~ grade + log(sqft\_living) + log(bedrooms +   
## 0.5) + exp(bathrooms) + waterfront + log(abs(lat - min(lat)) +   
## 0.5) + log(abs(long - min(long)) + 0.05) + (zipcode \* lat) +   
## log(view + 0.5) + condition + log(sqft\_above + 0.05) + log(sqft\_basement +   
## 0.05) + log(sqft\_lot15) + log(2015 - yr\_renovated + 1) +   
## (bedrooms \* bathrooms) + (log(grade) \* exp(condition)) +   
## (bedrooms \* log(sqft\_living)) + (view \* bedrooms), data = train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.06268 -0.14128 -0.00107 0.13625 1.09027   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 1.898e+04 1.613e+03 11.770 < 2e-16  
## grade 2.277e-01 3.257e-02 6.992 2.86e-12  
## log(sqft\_living) 3.127e-01 4.813e-02 6.497 8.54e-11  
## log(bedrooms + 0.5) 1.500e-01 8.016e-02 1.871 0.06138  
## exp(bathrooms) -1.897e-04 8.358e-04 -0.227 0.82047  
## waterfront 3.911e-01 3.811e-02 10.264 < 2e-16  
## log(abs(lat - min(lat)) + 0.5) 7.909e+00 1.631e-01 48.484 < 2e-16  
## log(abs(long - min(long)) + 0.05) -1.754e-01 8.090e-03 -21.684 < 2e-16  
## zipcode -1.890e-01 1.640e-02 -11.522 < 2e-16  
## lat -3.961e+02 3.392e+01 -11.679 < 2e-16  
## log(view + 0.5) 4.432e-02 2.637e-02 1.680 0.09290  
## condition 9.546e-02 1.117e-02 8.547 < 2e-16  
## log(sqft\_above + 0.05) 2.642e-01 2.323e-02 11.373 < 2e-16  
## log(sqft\_basement + 0.05) 8.928e-03 1.108e-03 8.060 8.38e-16  
## log(sqft\_lot15) -2.896e-02 4.605e-03 -6.289 3.31e-10  
## log(2015 - yr\_renovated + 1) -3.066e-02 2.106e-03 -14.556 < 2e-16  
## bedrooms -9.791e-02 1.053e-01 -0.930 0.35250  
## bathrooms -1.110e-01 1.809e-02 -6.137 8.69e-10  
## log(grade) -8.249e-01 2.561e-01 -3.221 0.00128  
## exp(condition) -7.185e-03 1.094e-03 -6.570 5.24e-11  
## view 6.997e-02 2.272e-02 3.080 0.00207  
## zipcode:lat 3.945e-03 3.449e-04 11.437 < 2e-16  
## bedrooms:bathrooms 2.941e-02 5.552e-03 5.297 1.20e-07  
## log(grade):exp(condition) 3.544e-03 5.437e-04 6.520 7.34e-11  
## log(sqft\_living):bedrooms -2.526e-03 1.303e-02 -0.194 0.84627  
## bedrooms:view -6.568e-03 3.641e-03 -1.804 0.07127  
##   
## (Intercept) \*\*\*  
## grade \*\*\*  
## log(sqft\_living) \*\*\*  
## log(bedrooms + 0.5) .   
## exp(bathrooms)   
## waterfront \*\*\*  
## log(abs(lat - min(lat)) + 0.5) \*\*\*  
## log(abs(long - min(long)) + 0.05) \*\*\*  
## zipcode \*\*\*  
## lat \*\*\*  
## log(view + 0.5) .   
## condition \*\*\*  
## log(sqft\_above + 0.05) \*\*\*  
## log(sqft\_basement + 0.05) \*\*\*  
## log(sqft\_lot15) \*\*\*  
## log(2015 - yr\_renovated + 1) \*\*\*  
## bedrooms   
## bathrooms \*\*\*  
## log(grade) \*\*   
## exp(condition) \*\*\*  
## view \*\*   
## zipcode:lat \*\*\*  
## bedrooms:bathrooms \*\*\*  
## log(grade):exp(condition) \*\*\*  
## log(sqft\_living):bedrooms   
## bedrooms:view .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.223 on 11232 degrees of freedom  
## Multiple R-squared: 0.7489, Adjusted R-squared: 0.7484   
## F-statistic: 1340 on 25 and 11232 DF, p-value: < 2.2e-16

## Let's compute the root mean square error

## computing RMSE for Model-05  
prediction05<-exp(predict(model05,interval='prediction',newdata=test))   
RMSE\_05=sqrt(sum((prediction05 - test$price)^2)/nrow(test))  
  
cat("RMSE for model03:",RMSE\_model03,"\nRMSE for model05:",RMSE\_05)

## RMSE for model03: 465951.2   
## RMSE for model05: 427364

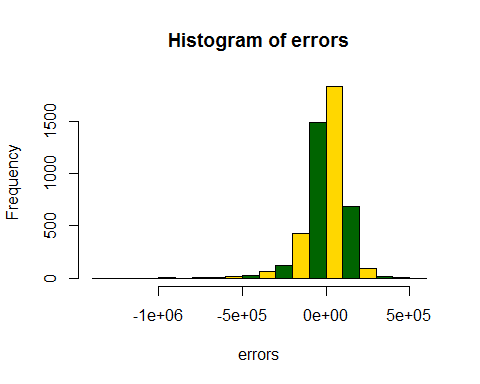
cat("Conclusion: RMSE for Model-03 is ",round(100\*(RMSE\_model03/RMSE\_05-1),2),"% more than Model-05, so here Model-05 is the clear winner.")

## Conclusion: RMSE for Model-03 is 9.03 % more than Model-05, so here Model-05 is the clear winner.

## Let's find the percentage of cases with less than 25% error.

## this process will give the confidence in the Model

errors <- prediction05[,'fit'] - test$price  
hist(errors,col=(c("gold","darkgreen")))



cat("Conclusion: we can see that Histogram of error shows that possibility of error are less, which gives confidence in the variable selection for the model with current set of data")

## Conclusion: we can see that Histogram of error shows that possibility of error are less, which gives confidence in the variable selection for the model with current set of data

rel\_change = 1 - ((test$price - abs(errors)) / test$price)  
##Now the percentage of cases with less than 25% error.  
pred25 = table(rel\_change<0.25)["TRUE"] / nrow(test)  
pred25

## TRUE   
## 0.7409863

cat("Conclusion: we can see 70%+ have 25% or less error.")

## Conclusion: we can see 70%+ have 25% or less error.

## 

## ################ END

## ###### stepByStep Analysis - Step -05

## ## Tip/Formula: what you will look for

## ######### when you buy a house

## ## enrich model with domain knowlege

## 

## 

## ###### stepByStep Analysis - Step -06

## ## Get most accurate

## ## Intercept & coefficient of Final Model

## ############# START

## 

## As above model is winner,

## let us run the model on different set of test and training data and find the best RMSE

## I Start with

## Training-dataset = 60%

## Testing-dataset = 40 % ,

## and increment Training-dataset by 1% data

## and continue till Training-dataset reach to 95%.

set.seed(1)  
#newhouseData <- subset(houseData, select = c(price,bathrooms,sqft\_living,grade,sqft\_above, bedrooms, waterfront, lat, long, view, condition, sqft\_basement, yr\_renovated, sqft\_lot15, zipcode))  
newhouseData <- houseData  
i=0.6  
storage <- list(c(), c(), c(),c())  
for(i in seq(from=0.60, to=0.95, by=0.01)){  
 rn\_train <- sample(nrow(newhouseData),floor(nrow(newhouseData)\*i))  
 train <- newhouseData[rn\_train,colnames(newhouseData)]  
 test <- newhouseData[-rn\_train,colnames(newhouseData)]  
 model <- lm(log(price)~grade+log(sqft\_living)+log(bedrooms+0.5)+exp(bathrooms)+waterfront+log(abs(lat-min(lat))+0.5)+log(abs(long-min(long))+0.05)+(zipcode\*lat)+log(view+0.5)+condition+log(sqft\_above+0.05)+log(sqft\_basement+0.05)+log(sqft\_lot15)+log(2015-yr\_renovated+1)+(bedrooms\*bathrooms)+(log(grade)\*exp(condition))+(bedrooms\*log(sqft\_living))+(view\*bedrooms),data=train)  
#summary(model)  
   
 prediction <- round(exp(predict(model,interval='prediction',newdata = test)),0)  
 train\_prediction = fitted(model)  
 train\_rmse = sqrt(sum((train\_prediction-train$price)^2)/nrow(train))  
 test\_rmse = sqrt(sum((prediction - test$price)^2)/nrow(test))  
   
 storage[[1]]<-c(storage[[1]],i)  
 storage[[2]]<-c(storage[[2]],test\_rmse)  
 storage[[3]]<-c(storage[[3]],train\_rmse)  
}  
  
##find the LM with minimun training error  
RMSE = storage[[3]]  
minimumVal = min(RMSE)  
minimumVal

## [1] 561425.3

indx = which(RMSE==min(RMSE))  
indx

## [1] 16

storage[[1]][indx]

## [1] 0.75

cat("\nConclusion: Minimum Training RMSE of Regression:",storage[[3]][indx],"\nRMSE of testing :",storage[[2]][indx], "\nTraining data Percentage:",storage[[1]][indx])

##   
## Conclusion: Minimum Training RMSE of Regression: 561425.3   
## RMSE of testing : 429032.6   
## Training data Percentage: 0.75

## 

## ###### stepByStep Analysis - Step -06

## ## Get most accurate

## ## Intercept & coefficient of Final Model

## ############# END

## 

## 

## ###### stepByStep Analysis - Step -07

## ##Goal: Find out % data having error

## ## less than 25% for predicted price

## ## Give evidence that model has

## ## higer accuracy - END

## 

## Now we come to a conclusion that 77% Training data provides the Minimum RMSE

## 1: SET training Data = 77% &

## 2: Get model with coeeficient & Intercept

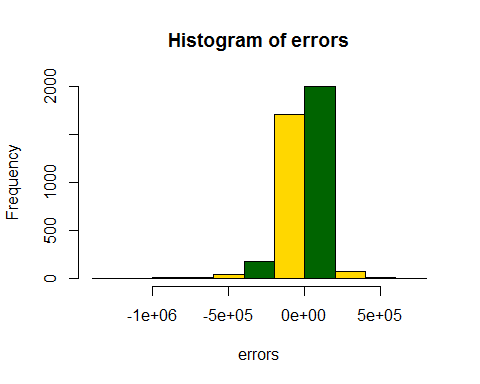
## 3: Draw the error of Histogram to get confidence with model

## 4: Find out how many data have less than 25% of error

rn\_train <- sample(nrow(newhouseData),floor(nrow(newhouseData)\*storage[[1]][indx]))  
train <- newhouseData[rn\_train,colnames(newhouseData)]  
test <- newhouseData[-rn\_train,colnames(newhouseData)]  
modelXGen <- lm(log(price)~grade+log(sqft\_living)+log(bedrooms+0.5)+exp(bathrooms)+waterfront+log(abs(lat-min(lat))+0.5)+log(abs(long-min(long))+0.05)+(zipcode\*lat)+log(view+0.5)+condition+log(sqft\_above+0.05)+log(sqft\_basement+0.05)+log(sqft\_lot15)+log(2015-yr\_renovated+1)+(bedrooms\*bathrooms)+(log(grade)\*exp(condition))+(bedrooms\*log(sqft\_living))+(view\*bedrooms),data=train)  
summary(modelXGen)

##   
## Call:  
## lm(formula = log(price) ~ grade + log(sqft\_living) + log(bedrooms +   
## 0.5) + exp(bathrooms) + waterfront + log(abs(lat - min(lat)) +   
## 0.5) + log(abs(long - min(long)) + 0.05) + (zipcode \* lat) +   
## log(view + 0.5) + condition + log(sqft\_above + 0.05) + log(sqft\_basement +   
## 0.05) + log(sqft\_lot15) + log(2015 - yr\_renovated + 1) +   
## (bedrooms \* bathrooms) + (log(grade) \* exp(condition)) +   
## (bedrooms \* log(sqft\_living)) + (view \* bedrooms), data = train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.04399 -0.14424 -0.00191 0.13870 1.19315   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 1.874e+04 1.555e+03 12.048 < 2e-16  
## grade 2.392e-01 3.177e-02 7.529 5.48e-14  
## log(sqft\_living) 3.515e-01 4.752e-02 7.397 1.48e-13  
## log(bedrooms + 0.5) 6.846e-02 7.950e-02 0.861 0.389226  
## exp(bathrooms) 1.668e-04 8.126e-04 0.205 0.837329  
## waterfront 4.662e-01 3.858e-02 12.084 < 2e-16  
## log(abs(lat - min(lat)) + 0.5) 7.966e+00 1.589e-01 50.147 < 2e-16  
## log(abs(long - min(long)) + 0.05) -1.799e-01 7.883e-03 -22.825 < 2e-16  
## zipcode -1.864e-01 1.582e-02 -11.788 < 2e-16  
## lat -3.908e+02 3.270e+01 -11.950 < 2e-16  
## log(view + 0.5) 8.606e-02 2.473e-02 3.480 0.000504  
## condition 8.973e-02 1.089e-02 8.241 < 2e-16  
## log(sqft\_above + 0.05) 2.681e-01 2.243e-02 11.954 < 2e-16  
## log(sqft\_basement + 0.05) 9.197e-03 1.070e-03 8.594 < 2e-16  
## log(sqft\_lot15) -3.094e-02 4.498e-03 -6.878 6.36e-12  
## log(2015 - yr\_renovated + 1) -2.787e-02 2.054e-03 -13.571 < 2e-16  
## bedrooms 6.028e-02 1.058e-01 0.570 0.568715  
## bathrooms -1.092e-01 1.767e-02 -6.181 6.58e-10  
## log(grade) -8.508e-01 2.500e-01 -3.403 0.000670  
## exp(condition) -5.351e-03 1.066e-03 -5.018 5.29e-07  
## view 2.647e-02 2.169e-02 1.220 0.222319  
## zipcode:lat 3.890e-03 3.326e-04 11.696 < 2e-16  
## bedrooms:bathrooms 2.820e-02 5.416e-03 5.206 1.97e-07  
## log(grade):exp(condition) 2.693e-03 5.317e-04 5.064 4.16e-07  
## log(sqft\_living):bedrooms -1.989e-02 1.300e-02 -1.530 0.126131  
## bedrooms:view -2.901e-03 3.646e-03 -0.796 0.426237  
##   
## (Intercept) \*\*\*  
## grade \*\*\*  
## log(sqft\_living) \*\*\*  
## log(bedrooms + 0.5)   
## exp(bathrooms)   
## waterfront \*\*\*  
## log(abs(lat - min(lat)) + 0.5) \*\*\*  
## log(abs(long - min(long)) + 0.05) \*\*\*  
## zipcode \*\*\*  
## lat \*\*\*  
## log(view + 0.5) \*\*\*  
## condition \*\*\*  
## log(sqft\_above + 0.05) \*\*\*  
## log(sqft\_basement + 0.05) \*\*\*  
## log(sqft\_lot15) \*\*\*  
## log(2015 - yr\_renovated + 1) \*\*\*  
## bedrooms   
## bathrooms \*\*\*  
## log(grade) \*\*\*  
## exp(condition) \*\*\*  
## view   
## zipcode:lat \*\*\*  
## bedrooms:bathrooms \*\*\*  
## log(grade):exp(condition) \*\*\*  
## log(sqft\_living):bedrooms   
## bedrooms:view   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.2245 on 12037 degrees of freedom  
## Multiple R-squared: 0.7479, Adjusted R-squared: 0.7474   
## F-statistic: 1429 on 25 and 12037 DF, p-value: < 2.2e-16

predictionXGen <- round(exp(predict(modelXGen,interval='prediction',newdata = test)),0)  
test\_rmseXGen = sqrt(sum((predictionXGen - test$price)^2)/nrow(test))  
errors <- predictionXGen[,'fit'] - test$price  
hist(errors,col=(c("gold","darkgreen")))



rel\_change = 1 - ((test$price - abs(errors)) / test$price)  
##Now the percentage of cases with less than 25% error.  
pred25 = table(rel\_change<0.25)["TRUE"] / nrow(test)  
pred25

## TRUE   
## 0.74857

cat("\nConclusion: Percent of data having less than 25% error:",pred25)

##   
## Conclusion: Percent of data having less than 25% error: 0.74857

## 

## ###### stepByStep Analysis - Step -07

## ##Goal: Find out % data having error

## ## less than 25% for predicted price

## ## Give evidence that model has

## ## higer accuracy - END

## 

## Now Write the file

#Now write the Real & Predicted price file for comparision  
predictionXGen <- round(exp(predict(modelXGen,newdata = test)),0)  
values <- (cbind("ID"=test$id,"Orginal Price"=test$price,"Predicted Price"=predictionXGen))  
write.csv(values, file = "RealPriceVsPredictedAsStaticticalAnalysis.csv", row.names=FALSE)

## 

## ###### stepByStep Analysis - End

## 

## 

## ####### stepAIC Analysis - START

## 

## Now Let perform feature selection using stepAIC

houseData <- read.csv("file:///G:/Ryerson-BigData/capstone-R/CKME-136/data/kc\_house\_data.csv")  
houseData$date<-NULL  
houseData$id<-NULL  
colnames(houseData)

## [1] "price" "bedrooms" "bathrooms" "sqft\_living"   
## [5] "sqft\_lot" "floors" "waterfront" "view"   
## [9] "condition" "grade" "sqft\_above" "sqft\_basement"  
## [13] "yr\_built" "yr\_renovated" "zipcode" "lat"   
## [17] "long" "sqft\_living15" "sqft\_lot15"

## Let us now use the forward selection algorithm using stepAIC.

full = lm(price~.,data=houseData)  
null = lm(price~1,data=houseData)  
stepF = stepAIC(null, scope=list(lower=null, upper=full), direction='forward', trace=TRUE)

## Start: AIC=553875.8  
## price ~ 1  
##   
## Df Sum of Sq RSS AIC  
## + sqft\_living 1 1.4356e+15 1.4773e+15 539204  
## + grade 1 1.2976e+15 1.6153e+15 541134  
## + sqft\_above 1 1.0682e+15 1.8447e+15 544004  
## + sqft\_living15 1 9.9816e+14 1.9148e+15 544810  
## + bathrooms 1 8.0329e+14 2.1096e+15 546904  
## + view 1 4.5978e+14 2.4531e+15 550165  
## + sqft\_basement 1 3.0544e+14 2.6075e+15 551484  
## + bedrooms 1 2.7696e+14 2.6360e+15 551718  
## + lat 1 2.7455e+14 2.6384e+15 551738  
## + waterfront 1 2.0668e+14 2.7062e+15 552287  
## + floors 1 1.9209e+14 2.7208e+15 552403  
## + yr\_renovated 1 4.6564e+13 2.8664e+15 553529  
## + sqft\_lot 1 2.3417e+13 2.8895e+15 553703  
## + sqft\_lot15 1 1.9801e+13 2.8931e+15 553730  
## + yr\_built 1 8.4977e+12 2.9044e+15 553815  
## + zipcode 1 8.2451e+12 2.9047e+15 553817  
## + condition 1 3.8514e+12 2.9091e+15 553849  
## + long 1 1.3624e+12 2.9116e+15 553868  
## <none> 2.9129e+15 553876  
##   
## Step: AIC=539203.5  
## price ~ sqft\_living  
##   
## Df Sum of Sq RSS AIC  
## + lat 1 2.1314e+14 1.2641e+15 535838  
## + view 1 1.2362e+14 1.3537e+15 537317  
## + grade 1 1.2132e+14 1.3560e+15 537353  
## + waterfront 1 1.1024e+14 1.3670e+15 537529  
## + yr\_built 1 9.2854e+13 1.3844e+15 537802  
## + long 1 6.6817e+13 1.4105e+15 538205  
## + bedrooms 1 4.0635e+13 1.4366e+15 538603  
## + zipcode 1 2.2858e+13 1.4544e+15 538868  
## + yr\_renovated 1 2.2405e+13 1.4549e+15 538875  
## + sqft\_living15 1 2.0109e+13 1.4572e+15 538909  
## + condition 1 1.7605e+13 1.4597e+15 538946  
## + sqft\_lot15 1 6.4407e+12 1.4708e+15 539111  
## + sqft\_lot 1 3.0113e+12 1.4743e+15 539161  
## + sqft\_above 1 1.2165e+12 1.4761e+15 539188  
## + sqft\_basement 1 1.2165e+12 1.4761e+15 539188  
## + floors 1 2.2991e+11 1.4770e+15 539202  
## + bathrooms 1 1.4719e+11 1.4771e+15 539203  
## <none> 1.4773e+15 539204  
##   
## Step: AIC=535838  
## price ~ sqft\_living + lat  
##   
## Df Sum of Sq RSS AIC  
## + view 1 1.2663e+14 1.1375e+15 533559  
## + waterfront 1 1.1646e+14 1.1477e+15 533751  
## + grade 1 8.8423e+13 1.1757e+15 534273  
## + yr\_built 1 5.1904e+13 1.2122e+15 534934  
## + long 1 3.6167e+13 1.2280e+15 535213  
## + bedrooms 1 3.2254e+13 1.2319e+15 535281  
## + condition 1 1.9095e+13 1.2450e+15 535511  
## + yr\_renovated 1 1.8897e+13 1.2452e+15 535515  
## + sqft\_living15 1 1.8325e+13 1.2458e+15 535524  
## + sqft\_lot15 1 1.2429e+12 1.2629e+15 535819  
## + zipcode 1 4.4621e+11 1.2637e+15 535832  
## <none> 1.2641e+15 535838  
## + sqft\_lot 1 1.0913e+11 1.2640e+15 535838  
## + sqft\_above 1 1.0387e+11 1.2640e+15 535838  
## + sqft\_basement 1 1.0387e+11 1.2640e+15 535838  
## + bathrooms 1 2.2942e+09 1.2641e+15 535840  
## + floors 1 2.9322e+07 1.2641e+15 535840  
##   
## Step: AIC=533558.7  
## price ~ sqft\_living + lat + view  
##   
## Df Sum of Sq RSS AIC  
## + grade 1 7.7085e+13 1.0604e+15 532044  
## + waterfront 1 4.8301e+13 1.0892e+15 532623  
## + yr\_built 1 2.9685e+13 1.1078e+15 532989  
## + bedrooms 1 2.0105e+13 1.1174e+15 533175  
## + long 1 1.8126e+13 1.1194e+15 533214  
## + condition 1 1.3259e+13 1.1242e+15 533307  
## + yr\_renovated 1 1.1033e+13 1.1265e+15 533350  
## + sqft\_living15 1 9.7773e+12 1.1277e+15 533374  
## + sqft\_above 1 5.6493e+12 1.1319e+15 533453  
## + sqft\_basement 1 5.6493e+12 1.1319e+15 533453  
## + sqft\_lot15 1 1.8222e+12 1.1357e+15 533526  
## + zipcode 1 1.3136e+12 1.1362e+15 533536  
## + floors 1 7.9084e+11 1.1367e+15 533546  
## + sqft\_lot 1 3.9207e+11 1.1371e+15 533553  
## + bathrooms 1 1.9270e+11 1.1373e+15 533557  
## <none> 1.1375e+15 533559  
##   
## Step: AIC=532044.1  
## price ~ sqft\_living + lat + view + grade  
##   
## Df Sum of Sq RSS AIC  
## + yr\_built 1 8.9146e+13 9.7128e+14 530148  
## + waterfront 1 5.0218e+13 1.0102e+15 530998  
## + condition 1 2.5997e+13 1.0344e+15 531510  
## + long 1 2.2309e+13 1.0381e+15 531587  
## + yr\_renovated 1 1.4312e+13 1.0461e+15 531752  
## + bedrooms 1 1.0398e+13 1.0500e+15 531833  
## + floors 1 3.9309e+12 1.0565e+15 531966  
## + bathrooms 1 2.2781e+12 1.0581e+15 532000  
## + sqft\_lot15 1 1.3272e+12 1.0591e+15 532019  
## + sqft\_lot 1 2.0910e+11 1.0602e+15 532042  
## + sqft\_above 1 1.3720e+11 1.0603e+15 532043  
## + sqft\_basement 1 1.3720e+11 1.0603e+15 532043  
## + sqft\_living15 1 1.1809e+11 1.0603e+15 532044  
## <none> 1.0604e+15 532044  
## + zipcode 1 7.8101e+10 1.0603e+15 532045  
##   
## Step: AIC=530148.3  
## price ~ sqft\_living + lat + view + grade + yr\_built  
##   
## Df Sum of Sq RSS AIC  
## + waterfront 1 5.0449e+13 9.2083e+14 528997  
## + bedrooms 1 1.1098e+13 9.6018e+14 529902  
## + zipcode 1 6.4623e+12 9.6481e+14 530006  
## + bathrooms 1 5.2656e+12 9.6601e+14 530033  
## + condition 1 4.2739e+12 9.6700e+14 530055  
## + long 1 2.8391e+12 9.6844e+14 530087  
## + yr\_renovated 1 2.3436e+12 9.6893e+14 530098  
## + floors 1 2.1809e+12 9.6910e+14 530102  
## + sqft\_above 1 2.1769e+12 9.6910e+14 530102  
## + sqft\_basement 1 2.1769e+12 9.6910e+14 530102  
## + sqft\_lot15 1 1.1384e+12 9.7014e+14 530125  
## + sqft\_living15 1 6.4656e+11 9.7063e+14 530136  
## + sqft\_lot 1 2.8898e+11 9.7099e+14 530144  
## <none> 9.7128e+14 530148  
##   
## Step: AIC=528997.4  
## price ~ sqft\_living + lat + view + grade + yr\_built + waterfront  
##   
## Df Sum of Sq RSS AIC  
## + bedrooms 1 9.0057e+12 9.1182e+14 528787  
## + zipcode 1 6.3395e+12 9.1449e+14 528850  
## + bathrooms 1 5.4031e+12 9.1543e+14 528872  
## + condition 1 4.4331e+12 9.1639e+14 528895  
## + long 1 2.5647e+12 9.1826e+14 528939  
## + floors 1 1.6628e+12 9.1917e+14 528960  
## + sqft\_above 1 1.4511e+12 9.1938e+14 528965  
## + sqft\_basement 1 1.4511e+12 9.1938e+14 528965  
## + yr\_renovated 1 1.2489e+12 9.1958e+14 528970  
## + sqft\_lot15 1 1.1644e+12 9.1966e+14 528972  
## + sqft\_living15 1 9.9743e+11 9.1983e+14 528976  
## + sqft\_lot 1 2.2143e+11 9.2061e+14 528994  
## <none> 9.2083e+14 528997  
##   
## Step: AIC=528787  
## price ~ sqft\_living + lat + view + grade + yr\_built + waterfront +   
## bedrooms  
##   
## Df Sum of Sq RSS AIC  
## + bathrooms 1 9.0102e+12 9.0281e+14 528574  
## + zipcode 1 6.9168e+12 9.0491e+14 528624  
## + condition 1 5.2627e+12 9.0656e+14 528664  
## + long 1 2.8458e+12 9.0898e+14 528721  
## + sqft\_lot15 1 1.9499e+12 9.0987e+14 528743  
## + floors 1 1.7197e+12 9.1010e+14 528748  
## + yr\_renovated 1 1.1626e+12 9.1066e+14 528761  
## + sqft\_above 1 1.1004e+12 9.1072e+14 528763  
## + sqft\_basement 1 1.1004e+12 9.1072e+14 528763  
## + sqft\_living15 1 8.3834e+11 9.1098e+14 528769  
## + sqft\_lot 1 5.6135e+11 9.1126e+14 528776  
## <none> 9.1182e+14 528787  
##   
## Step: AIC=528574.4  
## price ~ sqft\_living + lat + view + grade + yr\_built + waterfront +   
## bedrooms + bathrooms  
##   
## Df Sum of Sq RSS AIC  
## + zipcode 1 7.7118e+12 8.9510e+14 528391  
## + condition 1 4.8311e+12 8.9798e+14 528460  
## + long 1 2.0291e+12 9.0078e+14 528528  
## + sqft\_above 1 1.6201e+12 9.0119e+14 528538  
## + sqft\_basement 1 1.6201e+12 9.0119e+14 528538  
## + sqft\_living15 1 1.4832e+12 9.0133e+14 528541  
## + sqft\_lot15 1 1.4206e+12 9.0139e+14 528542  
## + yr\_renovated 1 4.2334e+11 9.0239e+14 528566  
## + floors 1 3.9100e+11 9.0242e+14 528567  
## + sqft\_lot 1 3.6234e+11 9.0245e+14 528568  
## <none> 9.0281e+14 528574  
##   
## Step: AIC=528391  
## price ~ sqft\_living + lat + view + grade + yr\_built + waterfront +   
## bedrooms + bathrooms + zipcode  
##   
## Df Sum of Sq RSS AIC  
## + long 1 9.6955e+12 8.8540e+14 528158  
## + condition 1 3.5280e+12 8.9157e+14 528308  
## + sqft\_lot15 1 2.1877e+12 8.9291e+14 528340  
## + sqft\_above 1 1.1628e+12 8.9394e+14 528365  
## + sqft\_basement 1 1.1628e+12 8.9394e+14 528365  
## + floors 1 1.0460e+12 8.9405e+14 528368  
## + sqft\_lot 1 7.2300e+11 8.9438e+14 528376  
## + sqft\_living15 1 4.3262e+11 8.9467e+14 528383  
## + yr\_renovated 1 3.8775e+11 8.9471e+14 528384  
## <none> 8.9510e+14 528391  
##   
## Step: AIC=528157.6  
## price ~ sqft\_living + lat + view + grade + yr\_built + waterfront +   
## bedrooms + bathrooms + zipcode + long  
##   
## Df Sum of Sq RSS AIC  
## + condition 1 3.2700e+12 8.8213e+14 528080  
## + sqft\_above 1 2.8185e+12 8.8259e+14 528091  
## + sqft\_basement 1 2.8185e+12 8.8259e+14 528091  
## + sqft\_living15 1 1.5701e+12 8.8383e+14 528121  
## + floors 1 8.8103e+11 8.8452e+14 528138  
## + sqft\_lot15 1 7.8011e+11 8.8462e+14 528141  
## + yr\_renovated 1 5.1267e+11 8.8489e+14 528147  
## <none> 8.8540e+14 528158  
## + sqft\_lot 1 8.0929e+10 8.8532e+14 528158  
##   
## Step: AIC=528079.6  
## price ~ sqft\_living + lat + view + grade + yr\_built + waterfront +   
## bedrooms + bathrooms + zipcode + long + condition  
##   
## Df Sum of Sq RSS AIC  
## + sqft\_above 1 3.6952e+12 8.7844e+14 527991  
## + sqft\_basement 1 3.6952e+12 8.7844e+14 527991  
## + sqft\_living15 1 1.7368e+12 8.8040e+14 528039  
## + floors 1 1.3208e+12 8.8081e+14 528049  
## + yr\_renovated 1 1.1069e+12 8.8103e+14 528055  
## + sqft\_lot15 1 7.9924e+11 8.8134e+14 528062  
## <none> 8.8213e+14 528080  
## + sqft\_lot 1 7.3301e+10 8.8206e+14 528080  
##   
## Step: AIC=527990.9  
## price ~ sqft\_living + lat + view + grade + yr\_built + waterfront +   
## bedrooms + bathrooms + zipcode + long + condition + sqft\_above  
##   
## Df Sum of Sq RSS AIC  
## + sqft\_living15 1 1.2623e+12 8.7718e+14 527962  
## + yr\_renovated 1 1.0677e+12 8.7737e+14 527967  
## + sqft\_lot15 1 8.9305e+11 8.7755e+14 527971  
## + floors 1 1.2603e+11 8.7831e+14 527990  
## + sqft\_lot 1 1.0887e+11 8.7833e+14 527990  
## <none> 8.7844e+14 527991  
##   
## Step: AIC=527961.8  
## price ~ sqft\_living + lat + view + grade + yr\_built + waterfront +   
## bedrooms + bathrooms + zipcode + long + condition + sqft\_above +   
## sqft\_living15  
##   
## Df Sum of Sq RSS AIC  
## + yr\_renovated 1 1.2135e+12 8.7596e+14 527934  
## + sqft\_lot15 1 9.2340e+11 8.7625e+14 527941  
## + floors 1 2.3644e+11 8.7694e+14 527958  
## + sqft\_lot 1 8.4109e+10 8.7709e+14 527962  
## <none> 8.7718e+14 527962  
##   
## Step: AIC=527933.9  
## price ~ sqft\_living + lat + view + grade + yr\_built + waterfront +   
## bedrooms + bathrooms + zipcode + long + condition + sqft\_above +   
## sqft\_living15 + yr\_renovated  
##   
## Df Sum of Sq RSS AIC  
## + sqft\_lot15 1 9.2931e+11 8.7503e+14 527913  
## + floors 1 1.8361e+11 8.7578e+14 527931  
## <none> 8.7596e+14 527934  
## + sqft\_lot 1 7.8513e+10 8.7589e+14 527934  
##   
## Step: AIC=527913  
## price ~ sqft\_living + lat + view + grade + yr\_built + waterfront +   
## bedrooms + bathrooms + zipcode + long + condition + sqft\_above +   
## sqft\_living15 + yr\_renovated + sqft\_lot15  
##   
## Df Sum of Sq RSS AIC  
## + sqft\_lot 1 2.8295e+11 8.7475e+14 527908  
## + floors 1 1.3147e+11 8.7490e+14 527912  
## <none> 8.7503e+14 527913  
##   
## Step: AIC=527908  
## price ~ sqft\_living + lat + view + grade + yr\_built + waterfront +   
## bedrooms + bathrooms + zipcode + long + condition + sqft\_above +   
## sqft\_living15 + yr\_renovated + sqft\_lot15 + sqft\_lot  
##   
## Df Sum of Sq RSS AIC  
## + floors 1 1.4017e+11 8.7461e+14 527907  
## <none> 8.7475e+14 527908  
##   
## Step: AIC=527906.5  
## price ~ sqft\_living + lat + view + grade + yr\_built + waterfront +   
## bedrooms + bathrooms + zipcode + long + condition + sqft\_above +   
## sqft\_living15 + yr\_renovated + sqft\_lot15 + sqft\_lot + floors  
##   
## Df Sum of Sq RSS AIC  
## <none> 8.7461e+14 527907

summary(stepF)

##   
## Call:  
## lm(formula = price ~ sqft\_living + lat + view + grade + yr\_built +   
## waterfront + bedrooms + bathrooms + zipcode + long + condition +   
## sqft\_above + sqft\_living15 + yr\_renovated + sqft\_lot15 +   
## sqft\_lot + floors, data = houseData)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1291725 -99229 -9739 77583 4333222   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.690e+06 2.931e+06 2.282 0.02249 \*   
## sqft\_living 1.501e+02 4.385e+00 34.227 < 2e-16 \*\*\*  
## lat 6.027e+05 1.073e+04 56.149 < 2e-16 \*\*\*  
## view 5.287e+04 2.140e+03 24.705 < 2e-16 \*\*\*  
## grade 9.589e+04 2.153e+03 44.542 < 2e-16 \*\*\*  
## yr\_built -2.620e+03 7.266e+01 -36.062 < 2e-16 \*\*\*  
## waterfront 5.830e+05 1.736e+04 33.580 < 2e-16 \*\*\*  
## bedrooms -3.577e+04 1.892e+03 -18.906 < 2e-16 \*\*\*  
## bathrooms 4.114e+04 3.254e+03 12.645 < 2e-16 \*\*\*  
## zipcode -5.824e+02 3.299e+01 -17.657 < 2e-16 \*\*\*  
## long -2.147e+05 1.313e+04 -16.349 < 2e-16 \*\*\*  
## condition 2.639e+04 2.351e+03 11.221 < 2e-16 \*\*\*  
## sqft\_above 3.113e+01 4.360e+00 7.139 9.71e-13 \*\*\*  
## sqft\_living15 2.168e+01 3.448e+00 6.289 3.26e-10 \*\*\*  
## yr\_renovated 1.981e+01 3.656e+00 5.420 6.03e-08 \*\*\*  
## sqft\_lot15 -3.826e-01 7.327e-02 -5.222 1.78e-07 \*\*\*  
## sqft\_lot 1.286e-01 4.792e-02 2.683 0.00729 \*\*   
## floors 6.690e+03 3.596e+03 1.860 0.06285 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 201200 on 21595 degrees of freedom  
## Multiple R-squared: 0.6997, Adjusted R-squared: 0.6995   
## F-statistic: 2960 on 17 and 21595 DF, p-value: < 2.2e-16

subsets = regsubsets(price~.,data=houseData,nbest=1,)

## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax,  
## force.in = force.in, : 1 linear dependencies found

## Reordering variables and trying again:

sub.sum = summary(subsets)  
as.data.frame(sub.sum$outmat)

## bedrooms bathrooms sqft\_living sqft\_lot floors waterfront view  
## 1 ( 1 ) \*   
## 2 ( 1 ) \*   
## 3 ( 1 ) \* \*  
## 4 ( 1 ) \*   
## 5 ( 1 ) \* \*   
## 6 ( 1 ) \* \* \*  
## 7 ( 1 ) \* \* \* \*  
## 8 ( 1 ) \* \* \* \* \*  
## 9 ( 1 ) \* \* \* \*  
## condition grade sqft\_above sqft\_basement yr\_built yr\_renovated  
## 1 ( 1 )   
## 2 ( 1 )   
## 3 ( 1 )   
## 4 ( 1 ) \* \*   
## 5 ( 1 ) \* \*   
## 6 ( 1 ) \* \*   
## 7 ( 1 ) \* \*   
## 8 ( 1 ) \* \*   
## 9 ( 1 ) \* \*   
## zipcode lat long sqft\_living15 sqft\_lot15  
## 1 ( 1 )   
## 2 ( 1 ) \*   
## 3 ( 1 ) \*   
## 4 ( 1 ) \*   
## 5 ( 1 ) \*   
## 6 ( 1 ) \*   
## 7 ( 1 ) \*   
## 8 ( 1 ) \*   
## 9 ( 1 ) \* \* \*

## Let us now use the backward selection algorithm using stepAIC.

full = lm(price~.,data=houseData)  
null = lm(price~1,data=houseData)  
stepF = stepAIC(full, direction= 'backward', trace=TRUE)

## Start: AIC=527906.5  
## price ~ bedrooms + bathrooms + sqft\_living + sqft\_lot + floors +   
## waterfront + view + condition + grade + sqft\_above + sqft\_basement +   
## yr\_built + yr\_renovated + zipcode + lat + long + sqft\_living15 +   
## sqft\_lot15  
##   
##   
## Step: AIC=527906.5  
## price ~ bedrooms + bathrooms + sqft\_living + sqft\_lot + floors +   
## waterfront + view + condition + grade + sqft\_above + yr\_built +   
## yr\_renovated + zipcode + lat + long + sqft\_living15 + sqft\_lot15  
##   
## Df Sum of Sq RSS AIC  
## <none> 8.7461e+14 527907  
## - floors 1 1.4017e+11 8.7475e+14 527908  
## - sqft\_lot 1 2.9164e+11 8.7490e+14 527912  
## - sqft\_lot15 1 1.1046e+12 8.7572e+14 527932  
## - yr\_renovated 1 1.1897e+12 8.7580e+14 527934  
## - sqft\_living15 1 1.6017e+12 8.7621e+14 527944  
## - sqft\_above 1 2.0640e+12 8.7668e+14 527955  
## - condition 1 5.0994e+12 8.7971e+14 528030  
## - bathrooms 1 6.4764e+12 8.8109e+14 528064  
## - long 1 1.0826e+13 8.8544e+14 528170  
## - zipcode 1 1.2626e+13 8.8724e+14 528214  
## - bedrooms 1 1.4476e+13 8.8909e+14 528259  
## - view 1 2.4720e+13 8.9933e+14 528507  
## - waterfront 1 4.5671e+13 9.2028e+14 529005  
## - sqft\_living 1 4.7447e+13 9.2206e+14 529046  
## - yr\_built 1 5.2669e+13 9.2728e+14 529168  
## - grade 1 8.0354e+13 9.5497e+14 529804  
## - lat 1 1.2769e+14 1.0023e+15 530850

summary(stepF)

##   
## Call:  
## lm(formula = price ~ bedrooms + bathrooms + sqft\_living + sqft\_lot +   
## floors + waterfront + view + condition + grade + sqft\_above +   
## yr\_built + yr\_renovated + zipcode + lat + long + sqft\_living15 +   
## sqft\_lot15, data = houseData)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1291725 -99229 -9739 77583 4333222   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.690e+06 2.931e+06 2.282 0.02249 \*   
## bedrooms -3.577e+04 1.892e+03 -18.906 < 2e-16 \*\*\*  
## bathrooms 4.114e+04 3.254e+03 12.645 < 2e-16 \*\*\*  
## sqft\_living 1.501e+02 4.385e+00 34.227 < 2e-16 \*\*\*  
## sqft\_lot 1.286e-01 4.792e-02 2.683 0.00729 \*\*   
## floors 6.690e+03 3.596e+03 1.860 0.06285 .   
## waterfront 5.830e+05 1.736e+04 33.580 < 2e-16 \*\*\*  
## view 5.287e+04 2.140e+03 24.705 < 2e-16 \*\*\*  
## condition 2.639e+04 2.351e+03 11.221 < 2e-16 \*\*\*  
## grade 9.589e+04 2.153e+03 44.542 < 2e-16 \*\*\*  
## sqft\_above 3.113e+01 4.360e+00 7.139 9.71e-13 \*\*\*  
## yr\_built -2.620e+03 7.266e+01 -36.062 < 2e-16 \*\*\*  
## yr\_renovated 1.981e+01 3.656e+00 5.420 6.03e-08 \*\*\*  
## zipcode -5.824e+02 3.299e+01 -17.657 < 2e-16 \*\*\*  
## lat 6.027e+05 1.073e+04 56.149 < 2e-16 \*\*\*  
## long -2.147e+05 1.313e+04 -16.349 < 2e-16 \*\*\*  
## sqft\_living15 2.168e+01 3.448e+00 6.289 3.26e-10 \*\*\*  
## sqft\_lot15 -3.826e-01 7.327e-02 -5.222 1.78e-07 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 201200 on 21595 degrees of freedom  
## Multiple R-squared: 0.6997, Adjusted R-squared: 0.6995   
## F-statistic: 2960 on 17 and 21595 DF, p-value: < 2.2e-16

subsets = regsubsets(price~.,data=houseData,nbest=1,)

## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax,  
## force.in = force.in, : 1 linear dependencies found

## Reordering variables and trying again:

sub.sum = summary(subsets)  
as.data.frame(sub.sum$outmat)

## bedrooms bathrooms sqft\_living sqft\_lot floors waterfront view  
## 1 ( 1 ) \*   
## 2 ( 1 ) \*   
## 3 ( 1 ) \* \*  
## 4 ( 1 ) \*   
## 5 ( 1 ) \* \*   
## 6 ( 1 ) \* \* \*  
## 7 ( 1 ) \* \* \* \*  
## 8 ( 1 ) \* \* \* \* \*  
## 9 ( 1 ) \* \* \* \*  
## condition grade sqft\_above sqft\_basement yr\_built yr\_renovated  
## 1 ( 1 )   
## 2 ( 1 )   
## 3 ( 1 )   
## 4 ( 1 ) \* \*   
## 5 ( 1 ) \* \*   
## 6 ( 1 ) \* \*   
## 7 ( 1 ) \* \*   
## 8 ( 1 ) \* \*   
## 9 ( 1 ) \* \*   
## zipcode lat long sqft\_living15 sqft\_lot15  
## 1 ( 1 )   
## 2 ( 1 ) \*   
## 3 ( 1 ) \*   
## 4 ( 1 ) \*   
## 5 ( 1 ) \*   
## 6 ( 1 ) \*   
## 7 ( 1 ) \*   
## 8 ( 1 ) \*   
## 9 ( 1 ) \* \* \*

sapply(houseData, is.numeric)

## price bedrooms bathrooms sqft\_living sqft\_lot   
## TRUE TRUE TRUE TRUE TRUE   
## floors waterfront view condition grade   
## TRUE TRUE TRUE TRUE TRUE   
## sqft\_above sqft\_basement yr\_built yr\_renovated zipcode   
## TRUE TRUE TRUE TRUE TRUE   
## lat long sqft\_living15 sqft\_lot15   
## TRUE TRUE TRUE TRUE

houseDatas <- houseData[ , sapply(houseData, is.numeric)]  
cor(houseDatas)

## price bedrooms bathrooms sqft\_living  
## price 1.00000000 0.308349598 0.52513751 0.70203505  
## bedrooms 0.30834960 1.000000000 0.51588364 0.57667069  
## bathrooms 0.52513751 0.515883638 1.00000000 0.75466528  
## sqft\_living 0.70203505 0.576670693 0.75466528 1.00000000  
## sqft\_lot 0.08966086 0.031703243 0.08773966 0.17282566  
## floors 0.25679389 0.175428935 0.50065317 0.35394929  
## waterfront 0.26636943 -0.006582479 0.06374363 0.10381782  
## view 0.39729349 0.079531852 0.18773702 0.28461119  
## condition 0.03636179 0.028472104 -0.12498193 -0.05875259  
## grade 0.66743426 0.356966725 0.66498253 0.76270448  
## sqft\_above 0.60556730 0.477600161 0.68534248 0.87659660  
## sqft\_basement 0.32381602 0.303093375 0.28377003 0.43504297  
## yr\_built 0.05401153 0.154178069 0.50601944 0.31804877  
## yr\_renovated 0.12643379 0.018840823 0.05073898 0.05536293  
## zipcode -0.05320285 -0.152668487 -0.20386627 -0.19943004  
## lat 0.30700348 -0.008931010 0.02457295 0.05252946  
## long 0.02162624 0.129472975 0.22304184 0.24022330  
## sqft\_living15 0.58537890 0.391637524 0.56863429 0.75642026  
## sqft\_lot15 0.08244715 0.029244224 0.08717536 0.18328555  
## sqft\_lot floors waterfront view  
## price 0.089660861 0.256793888 0.266369434 0.397293488  
## bedrooms 0.031703243 0.175428935 -0.006582479 0.079531852  
## bathrooms 0.087739662 0.500653173 0.063743629 0.187737024  
## sqft\_living 0.172825661 0.353949290 0.103817818 0.284611186  
## sqft\_lot 1.000000000 -0.005200991 0.021603683 0.074710106  
## floors -0.005200991 1.000000000 0.023698320 0.029443820  
## waterfront 0.021603683 0.023698320 1.000000000 0.401857351  
## view 0.074710106 0.029443820 0.401857351 1.000000000  
## condition -0.008958250 -0.263767946 0.016653157 0.045989737  
## grade 0.113621124 0.458182514 0.082774914 0.251320585  
## sqft\_above 0.183512281 0.523884710 0.072074592 0.167649344  
## sqft\_basement 0.015286202 -0.245704542 0.080587939 0.276946579  
## yr\_built 0.053080367 0.489319425 -0.026161086 -0.053439851  
## yr\_renovated 0.007643505 0.006338401 0.092884837 0.103917288  
## zipcode -0.129574486 -0.059120642 0.030284728 0.084826917  
## lat -0.085682788 0.049614131 -0.014273776 0.006156732  
## long 0.229520859 0.125419028 -0.041910200 -0.078399712  
## sqft\_living15 0.144608174 0.279885265 0.086463136 0.280439082  
## sqft\_lot15 0.718556752 -0.011269187 0.030703283 0.072574568  
## condition grade sqft\_above sqft\_basement  
## price 0.036361789 0.66743426 0.6055672984 0.32381602  
## bedrooms 0.028472104 0.35696673 0.4776001614 0.30309338  
## bathrooms -0.124981933 0.66498253 0.6853424759 0.28377003  
## sqft\_living -0.058752587 0.76270448 0.8765965987 0.43504297  
## sqft\_lot -0.008958250 0.11362112 0.1835122809 0.01528620  
## floors -0.263767946 0.45818251 0.5238847103 -0.24570454  
## waterfront 0.016653157 0.08277491 0.0720745917 0.08058794  
## view 0.045989737 0.25132058 0.1676493441 0.27694658  
## condition 1.000000000 -0.14467367 -0.1582136164 0.17410491  
## grade -0.144673671 1.00000000 0.7559229376 0.16839182  
## sqft\_above -0.158213616 0.75592294 1.0000000000 -0.05194331  
## sqft\_basement 0.174104914 0.16839182 -0.0519433068 1.00000000  
## yr\_built -0.361416562 0.44696320 0.4238983517 -0.13312410  
## yr\_renovated -0.060617787 0.01441428 0.0232846879 0.07132290  
## zipcode 0.003025524 -0.18486209 -0.2611899765 0.07484461  
## lat -0.014941006 0.11408406 -0.0008164986 0.11053796  
## long -0.106500448 0.19837215 0.3438030175 -0.14476477  
## sqft\_living15 -0.092824268 0.71320209 0.7318702924 0.20035498  
## sqft\_lot15 -0.003405523 0.11924790 0.1940498619 0.01727618  
## yr\_built yr\_renovated zipcode lat  
## price 0.05401153 0.126433793 -0.053202854 0.3070034800  
## bedrooms 0.15417807 0.018840823 -0.152668487 -0.0089310097  
## bathrooms 0.50601944 0.050738978 -0.203866274 0.0245729528  
## sqft\_living 0.31804877 0.055362927 -0.199430043 0.0525294622  
## sqft\_lot 0.05308037 0.007643505 -0.129574486 -0.0856827882  
## floors 0.48931942 0.006338401 -0.059120642 0.0496141310  
## waterfront -0.02616109 0.092884837 0.030284728 -0.0142737756  
## view -0.05343985 0.103917288 0.084826917 0.0061567321  
## condition -0.36141656 -0.060617787 0.003025524 -0.0149410064  
## grade 0.44696320 0.014414281 -0.184862093 0.1140840571  
## sqft\_above 0.42389835 0.023284688 -0.261189977 -0.0008164986  
## sqft\_basement -0.13312410 0.071322902 0.074844608 0.1105379580  
## yr\_built 1.00000000 -0.224873518 -0.346869178 -0.1481224021  
## yr\_renovated -0.22487352 1.000000000 0.064357057 0.0293976092  
## zipcode -0.34686918 0.064357057 1.000000000 0.2670479500  
## lat -0.14812240 0.029397609 0.267047950 1.0000000000  
## long 0.40935620 -0.068372369 -0.564071606 -0.1355117836  
## sqft\_living15 0.32622890 -0.002672555 -0.279032997 0.0488579321  
## sqft\_lot15 0.07095793 0.007853765 -0.147221069 -0.0864188072  
## long sqft\_living15 sqft\_lot15  
## price 0.02162624 0.585378904 0.082447153  
## bedrooms 0.12947298 0.391637524 0.029244224  
## bathrooms 0.22304184 0.568634290 0.087175361  
## sqft\_living 0.24022330 0.756420259 0.183285551  
## sqft\_lot 0.22952086 0.144608174 0.718556752  
## floors 0.12541903 0.279885265 -0.011269187  
## waterfront -0.04191020 0.086463136 0.030703283  
## view -0.07839971 0.280439082 0.072574568  
## condition -0.10650045 -0.092824268 -0.003405523  
## grade 0.19837215 0.713202093 0.119247897  
## sqft\_above 0.34380302 0.731870292 0.194049862  
## sqft\_basement -0.14476477 0.200354983 0.017276181  
## yr\_built 0.40935620 0.326228900 0.070957926  
## yr\_renovated -0.06837237 -0.002672555 0.007853765  
## zipcode -0.56407161 -0.279032997 -0.147221069  
## lat -0.13551178 0.048857932 -0.086418807  
## long 1.00000000 0.334604984 0.254451288  
## sqft\_living15 0.33460498 1.000000000 0.183191749  
## sqft\_lot15 0.25445129 0.183191749 1.000000000

## Using Step AIC we got below variables needs to part of model

## 1:sqft\_living

## 2:lat

## 3:grade

## 4:yr\_built

## 5:waterfront

## 6:bedrooms

## 7:bathrooms

## 8:zipcode

## 9:long

#since floor is having very less contribution to price.we are not including floor in our model.  
houseData$floors<-NULL  
houseData$date<-NULL  
colnames(houseData)

## [1] "price" "bedrooms" "bathrooms" "sqft\_living"   
## [5] "sqft\_lot" "waterfront" "view" "condition"   
## [9] "grade" "sqft\_above" "sqft\_basement" "yr\_built"   
## [13] "yr\_renovated" "zipcode" "lat" "long"   
## [17] "sqft\_living15" "sqft\_lot15"

newhouseDataAIC <- subset(houseData, select = c(price,sqft\_living,lat,view,grade,yr\_built,waterfront,bedrooms,bathrooms,zipcode,long))  
  
set.seed(1)  
i=0.6  
storageAIC <- list(c(), c(), c(),c())  
for(i in seq(from=0.6, to=0.9, by=0.01)){  
 rn\_train <- sample(nrow(newhouseDataAIC),floor(nrow(newhouseDataAIC)\*i))  
 train <- newhouseDataAIC[rn\_train,colnames(newhouseDataAIC)]  
 test <- newhouseDataAIC[-rn\_train,colnames(newhouseDataAIC)]  
 model<-lm(price~sqft\_living+lat+view+grade+yr\_built+waterfront+bedrooms+bathrooms+zipcode+long,data = train)  
 prediction <- predict(model,interval='prediction',newdata = test)  
 train\_prediction = fitted(model)  
 train\_rmse = sqrt(sum((train\_prediction-train$price)^2)/nrow(train))  
 test\_rmse = sqrt(sum((prediction - test$price)^2)/nrow(test))  
 storageAIC[[1]]<-c(storageAIC[[1]],i)  
 storageAIC[[2]]<-c(storageAIC[[2]],test\_rmse)  
 storageAIC[[3]]<-c(storageAIC[[3]],train\_rmse)  
   
}  
  
##find the LM with minimun training error  
RMSE = storageAIC[[3]]  
minimumVal = min(RMSE)  
minimumVal

## [1] 196954.3

indxAIC = which(RMSE==min(RMSE))  
indxAIC

## [1] 14

storageAIC[[1]][indxAIC]

## [1] 0.73

cat("\nStepAIC Finding: Minimum Training RMSE of Regression:",storageAIC[[3]][indxAIC],"\nRMSE of testing :",storageAIC[[2]][indxAIC], "\nTraining data Percentage:",storageAIC[[1]][indxAIC])

##   
## StepAIC Finding: Minimum Training RMSE of Regression: 196954.3   
## RMSE of testing : 663350.1   
## Training data Percentage: 0.73

## Now we come to a conclusion that 73% Training data provides the Minimum RMSE

## 1: SET training Data = 73% &

## 2: Get model with coeeficient & Intercept

## 3: Draw the error of Histogram to get confidence with model

## 4: Find out how many data have less than 25% of error

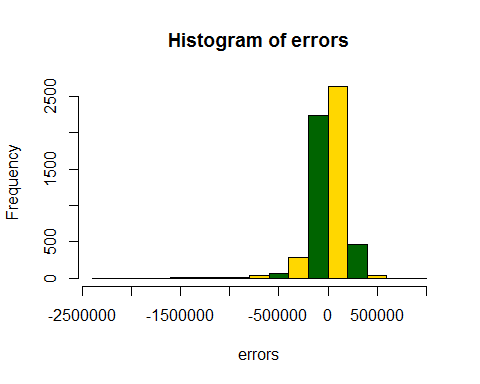
set.seed(1)  
rn\_train <- sample(nrow(newhouseDataAIC),floor(nrow(newhouseDataAIC)\*storage[[1]][indxAIC]))  
train <- newhouseDataAIC[rn\_train,colnames(newhouseDataAIC)]  
test <- newhouseDataAIC[-rn\_train,colnames(newhouseDataAIC)]  
modelXGenAIC <- lm(price~sqft\_living+lat+view+grade+yr\_built+waterfront+bedrooms+bathrooms+zipcode+long,data = train)  
summary(modelXGenAIC)

##   
## Call:  
## lm(formula = price ~ sqft\_living + lat + view + grade + yr\_built +   
## waterfront + bedrooms + bathrooms + zipcode + long, data = train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1242635 -99486 -9406 78454 4345822   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.581e+07 3.267e+06 4.839 1.32e-06 \*\*\*  
## sqft\_living 1.755e+02 3.588e+00 48.918 < 2e-16 \*\*\*  
## lat 5.915e+05 1.258e+04 47.026 < 2e-16 \*\*\*  
## view 5.297e+04 2.470e+03 21.447 < 2e-16 \*\*\*  
## grade 1.039e+05 2.362e+03 43.983 < 2e-16 \*\*\*  
## yr\_built -2.806e+03 7.497e+01 -37.425 < 2e-16 \*\*\*  
## waterfront 5.656e+05 2.059e+04 27.463 < 2e-16 \*\*\*  
## bedrooms -3.394e+04 2.191e+03 -15.495 < 2e-16 \*\*\*  
## bathrooms 4.063e+04 3.642e+03 11.154 < 2e-16 \*\*\*  
## zipcode -6.284e+02 3.838e+01 -16.375 < 2e-16 \*\*\*  
## long -1.850e+05 1.491e+04 -12.414 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 203900 on 15766 degrees of freedom  
## Multiple R-squared: 0.6878, Adjusted R-squared: 0.6876   
## F-statistic: 3473 on 10 and 15766 DF, p-value: < 2.2e-16

predictionXGenAIC <- predict(modelXGenAIC,interval='prediction',newdata = test)  
test\_rmseXGenAIC = sqrt(sum((predictionXGen - test$price)^2)/nrow(test))

## Warning in predictionXGen - test$price: longer object length is not a  
## multiple of shorter object length

errors <- predictionXGenAIC[,'fit'] - test$price  
hist(errors,col=(c("gold","darkgreen")))



rel\_change = 1 - ((test$price - abs(errors)) / test$price)  
##Now the percentage of cases with less than 25% error.  
pred25AIC = table(rel\_change<0.25)["TRUE"] / nrow(test)  
pred25AIC

## TRUE   
## 0.6201165

cat("\nConclusion:percent of data having less than 25% error:",pred25)

##   
## Conclusion:percent of data having less than 25% error: 0.74857

## 

## ####### stepAIC Analysis - END

## 

## 

## ####### Boruta Analysis - START

## 

# USING BORUTA for features selection and finally making model with the selected attributes

set.seed(1)  
boruta.train <- Boruta(price ~., data = houseData , doTrace = 2,ntree = 500)

## 1. run of importance source...

## 2. run of importance source...

## 3. run of importance source...

## 4. run of importance source...

## 5. run of importance source...

## 6. run of importance source...

## 7. run of importance source...

## 8. run of importance source...

## 9. run of importance source...

## 10. run of importance source...

## 11. run of importance source...

## After 11 iterations, +8 mins:

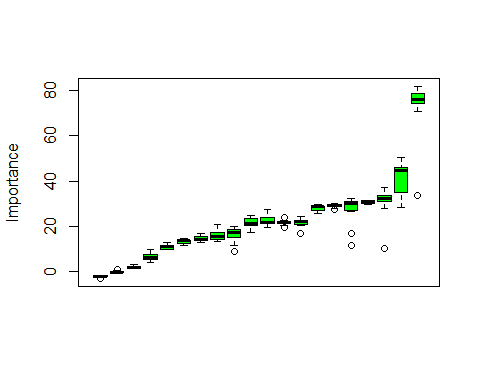
## confirmed 17 attributes: bathrooms, bedrooms, condition, grade, lat and 12 more;

## no more attributes left.

print(boruta.train)

## Boruta performed 11 iterations in 8.023046 mins.  
## 17 attributes confirmed important: bathrooms, bedrooms,  
## condition, grade, lat and 12 more;  
## No attributes deemed unimportant.

plot(boruta.train, xlab = "", xaxt = "n")



Boruta.Short <- Boruta(price ~ ., data = houseData, maxRuns = 12)

# Start calculating RMSE

newhouseDataBoruta <- houseData  
  
set.seed(1)  
i=0.6  
storageBoruta <- list(c(), c(), c(),c())  
for(i in seq(from=0.6, to=0.9, by=0.01)){  
 rn\_train <- sample(nrow(newhouseDataBoruta),floor(nrow(newhouseDataBoruta)\*i))  
 train <- newhouseDataBoruta[rn\_train,colnames(newhouseDataBoruta)]  
 test <- newhouseDataBoruta[-rn\_train,colnames(newhouseDataBoruta)]  
 model <- lm(formula = price~.,data=train)  
 prediction <- predict(model,interval='prediction',newdata = test)  
 train\_prediction = fitted(model)  
 train\_rmse = sqrt(sum((train\_prediction-train$price)^2)/nrow(train))  
 test\_rmse = sqrt(sum((prediction - test$price)^2)/nrow(test))  
 storageBoruta[[1]]<-c(storageBoruta[[1]],i)  
 storageBoruta[[2]]<-c(storageBoruta[[2]],test\_rmse)  
 storageBoruta[[3]]<-c(storageBoruta[[3]],train\_rmse)  
}

## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading  
  
## Warning in predict.lm(model, interval = "prediction", newdata = test):  
## prediction from a rank-deficient fit may be misleading

##find the LM with minimun training error  
RMSE = storageBoruta[[3]]  
minimumVal = min(RMSE)  
minimumVal

## [1] 195597.5

indxBoruta = which(RMSE==min(RMSE))  
indxBoruta

## [1] 14

storageBoruta[[1]][indxBoruta]

## [1] 0.73

cat("\nBourta Finding: Minimum Training RMSE of Regression:",storageBoruta[[3]][indxBoruta],"\nRMSE of testing :",storageBoruta[[2]][indxBoruta], "\nTraining data Percentage:",storageBoruta[[1]][indxBoruta])

##   
## Bourta Finding: Minimum Training RMSE of Regression: 195597.5   
## RMSE of testing : 659735.6   
## Training data Percentage: 0.73

## Now we come to a conclusion that 73% Training data provides the Minimum RMSE

## 1: SET training Data = 73% &

## 2: Get model with coeeficient & Intercept

## 3: Draw the error of Histogram to get confidence with model

## 4: Find out how many data have less than 25% of error

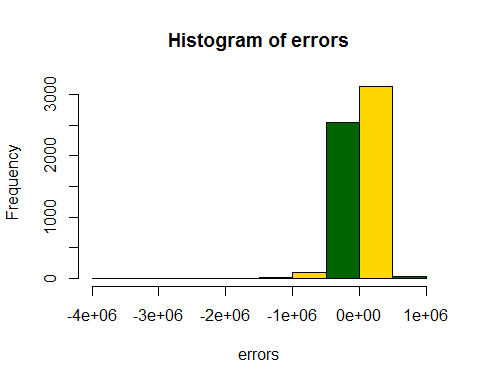
rn\_train <- sample(nrow(newhouseDataBoruta),floor(nrow(newhouseDataBoruta)\*storageBoruta[[1]][indxBoruta]))  
train <- newhouseDataBoruta[rn\_train,colnames(newhouseDataBoruta)]  
test <- newhouseDataBoruta[-rn\_train,colnames(newhouseDataBoruta)]  
modelXGenBoruta <- lm(price~.,data = train)  
summary(modelXGenBoruta)

##   
## Call:  
## lm(formula = price ~ ., data = train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1244403 -98183 -9044 76993 4349275   
##   
## Coefficients: (1 not defined because of singularities)  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 7.204e+06 3.313e+06 2.174 0.0297 \*   
## bedrooms -3.877e+04 2.266e+03 -17.110 < 2e-16 \*\*\*  
## bathrooms 4.630e+04 3.609e+03 12.831 < 2e-16 \*\*\*  
## sqft\_living 1.462e+02 4.823e+00 30.309 < 2e-16 \*\*\*  
## sqft\_lot 1.052e-01 5.226e-02 2.013 0.0441 \*   
## waterfront 5.261e+05 1.972e+04 26.679 < 2e-16 \*\*\*  
## view 5.787e+04 2.477e+03 23.361 < 2e-16 \*\*\*  
## condition 2.668e+04 2.697e+03 9.890 < 2e-16 \*\*\*  
## grade 9.614e+04 2.464e+03 39.012 < 2e-16 \*\*\*  
## sqft\_above 3.039e+01 4.512e+00 6.736 1.69e-11 \*\*\*  
## sqft\_basement NA NA NA NA   
## yr\_built -2.607e+03 8.157e+01 -31.958 < 2e-16 \*\*\*  
## yr\_renovated 2.068e+01 4.220e+00 4.900 9.70e-07 \*\*\*  
## zipcode -5.749e+02 3.778e+01 -15.217 < 2e-16 \*\*\*  
## lat 6.005e+05 1.235e+04 48.634 < 2e-16 \*\*\*  
## long -2.053e+05 1.499e+04 -13.698 < 2e-16 \*\*\*  
## sqft\_living15 2.033e+01 3.939e+00 5.162 2.48e-07 \*\*\*  
## sqft\_lot15 -3.268e-01 8.131e-02 -4.019 5.87e-05 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 197700 on 15760 degrees of freedom  
## Multiple R-squared: 0.701, Adjusted R-squared: 0.7007   
## F-statistic: 2309 on 16 and 15760 DF, p-value: < 2.2e-16

predictionXGenBoruta <- predict(modelXGenBoruta,interval='prediction',newdata = test)

## Warning in predict.lm(modelXGenBoruta, interval = "prediction", newdata =  
## test): prediction from a rank-deficient fit may be misleading

test\_rmseXGenBoruta = sqrt(sum((predictionXGenBoruta - test$price)^2)/nrow(test))  
errors <- predictionXGenBoruta[,'fit'] - test$price  
hist(errors,col=(c("gold","darkgreen")))



rel\_change = 1 - ((test$price - abs(errors)) / test$price)  
##Now the percentage of cases with less than 25% error.  
pred25Boruta = table(rel\_change<0.25)["TRUE"] / nrow(test)  
pred25Boruta

## TRUE   
## 0.6101782

cat("\nConclusion:percent of data having less than 25% error:",pred25Boruta)

##   
## Conclusion:percent of data having less than 25% error: 0.6101782

## 

## ####### Boruta Analysis - END

## 

## 

## ###### Summary Of

## ###### StepAIC

## ########## Vs

## ###### Boruta

## ######### Vs

## ###### Step By Step Analysis

## ################# START

## 

## Below is the Summary of three models

## 1: Model by StepByStep Analysis ##2: stepAIC ##3: Boruta

cat("\nModel Category\t\t\tRMSE\t\t %ofData-havig-less-than 25% error","\nModel by StepByStep Analysis\t\t\t",test\_rmseXGen,"\t\t",pred25,"\nstepAIC\t\t\t",test\_rmseXGenAIC,"\t\t",pred25AIC,"\nBoruta\t\t\t",test\_rmseXGenBoruta,"\t\t",pred25Boruta,"\n\nConclusion: Model by StepByStep Analysis with RMSE",test\_rmseXGen,"%ofData-havig-less-than 25% error",pred25," is Winner.")

##   
## Model Category RMSE %ofData-havig-less-than 25% error   
## Model by StepByStep Analysis 416828.4 0.74857   
## stepAIC 426436 0.6201165   
## Boruta 658940.6 0.6101782   
##   
## Conclusion: Model by StepByStep Analysis with RMSE 416828.4 %ofData-havig-less-than 25% error 0.74857 is Winner.

## 

## ###### Summary Of

## ###### StepAIC

## ########## Vs

## ###### Boruta

## ######### Vs

## ###### Step By Step Analysis

## ################# END

## 

## 

## ###### Step By Step Analysis

## ###### Different Model Comparison

## ########### at

## ###### best data partition

## ############ START

## 

## Now let us verify RMSE for each Model derived due to stepByStep analysis

## I would like to find out the winner among all the model generated due to StepByStep is winner

#Get The data divided   
set.seed(1)  
rn\_train <- sample(nrow(newhouseData),floor(nrow(newhouseData)\*storage[[1]][indx]))  
train <- newhouseData[rn\_train,colnames(newhouseData)]  
test <- newhouseData[-rn\_train,colnames(newhouseData)]  
  
#Model01-Start  
model01 <- lm(data=train,log(price)~grade)  
predic\_model01<-exp(predict(model01,interval='prediction',newdata=test))   
RMSE\_model01=sqrt(sum((predic\_model01 - test$price)^2)/nrow(test))   
  
#Model02-Start  
model02<-lm(log(price)~log(sqft\_living)+bedrooms+bathrooms+grade+waterfront,data=train)  
predic\_model02<-exp(predict(model02,interval='prediction',newdata=test))   
RMSE\_model02=sqrt(sum((predic\_model02 - test$price)^2)/nrow(test))   
  
  
#Model03-Start  
model03<-lm(log(price)~log(sqft\_living)+bedrooms+bathrooms+grade+waterfront+yr\_built+lat,data=train)  
predic\_model03<-exp(predict(model03,interval='prediction',newdata=test))  
RMSE\_model03=sqrt(sum((predic\_model03 - test$price)^2)/nrow(test))  
  
##modelXGen Start  
modelXGen <- lm(log(price)~grade+log(sqft\_living)+log(bedrooms+0.5)+exp(bathrooms)+waterfront+log(abs(lat-min(lat))+0.5)+log(abs(long-min(long))+0.05)+(zipcode\*lat)+log(view+0.5)+condition+log(sqft\_above+0.05)+log(sqft\_basement+0.05)+log(sqft\_lot15)+log(2015-yr\_renovated+1)+(bedrooms\*bathrooms)+(log(grade)\*exp(condition))+(bedrooms\*log(sqft\_living))+(view\*bedrooms),data=train)  
predictionXGen <- round(exp(predict(modelXGen,interval='prediction',newdata = test)),0)  
test\_rmseXGen = sqrt(sum((predictionXGen - test$price)^2)/nrow(test))  
  
  
cat("Below are the finding when train-data:",storage[[1]][indx]," and Testing-Data:", (1-storage[[1]][indx]))

## Below are the finding when train-data: 0.75 and Testing-Data: 0.25

cat("\nRMSE for model01:",RMSE\_model01,"\nRMSE for model02:",RMSE\_model02,"\nRMSE for model03:",RMSE\_model03,"\nRMSE for modelXGen:", test\_rmseXGen)

##   
## RMSE for model01: 655319.5   
## RMSE for model02: 615577.5   
## RMSE for model03: 467134.3   
## RMSE for modelXGen: 428520.4

cat("\nFindings: RMSE for Model-01 is ",round(100\*(RMSE\_model01/RMSE\_model02-1),2),"% more than Model-02. So Model-02 predicts the prices better.")

##   
## Findings: RMSE for Model-01 is 6.46 % more than Model-02. So Model-02 predicts the prices better.

cat("\nFindings: RMSE for Model-02 is ",round(100\*(RMSE\_model02/RMSE\_model03-1),2),"% more than Model-03. So Model-03 predicts the prices better.")

##   
## Findings: RMSE for Model-02 is 31.78 % more than Model-03. So Model-03 predicts the prices better.

cat("\n\n Conclusion: RMSE for Model-03 is ",round(100\*(RMSE\_model03/test\_rmseXGen-1),2),"% more than Model-XGen.\n\nSo Model-xGen predicts the prices better even at ", storage[[1]][indx] , " percentage of training data.")

##   
##   
## Conclusion: RMSE for Model-03 is 9.01 % more than Model-XGen.  
##   
## So Model-xGen predicts the prices better even at 0.75 percentage of training data.

## 

## ###### Step By Step Analysis

## ###### Different Model Comparison

## ########### at

## ###### best data partition

## ############ END

## 