#library(lubridate)  
#library(lazyeval)  
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\RtmpyUyWb8\downloaded\_packages

library(ggplot2)  
library(GGally)

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], minP, maxP))  
}

analysis <- function(houseData, i, labels, plotLog){  
 plot(houseData$price~houseData[,i],main = labels[1],xlab = labels[2],ylab = labels[3], col=(c("gold","darkgreen")))  
   
 plot(houseData[,i],log(houseData$price), main=cat(labels[1]), xlab=cat(labels[2]), ylab=cat("Log of ",labels[3]), col=(c("gold","darkgreen")))  
   
 if(plotLog=='Y'){  
 plot(log(houseData[,i]),log(houseData$price), main=cat("Log ",labels[1]), xlab=cat("Log of ",labels[2]), ylab=cat("Log of ",labels[3]), col=(c("gold","darkgreen")))  
 }  
   
 hist(houseData[,i],main = labels[1],xlab = labels[2],ylab = labels[3],col=(c("gold","darkgreen")))  
  
 boxplot(houseData[,i],main = labels[1],col=(c("gold","darkgreen")))  
   
 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)

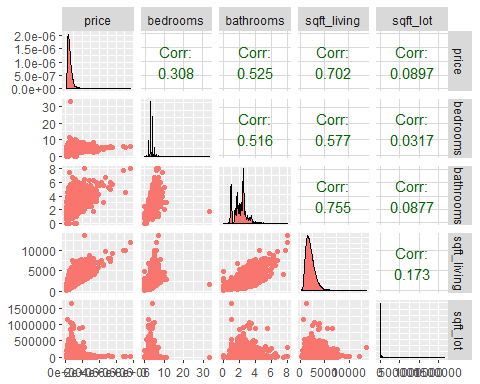
## [1] "id" "date" "price" "bedrooms"   
## [5] "bathrooms" "sqft\_living" "sqft\_lot" "floors"   
## [9] "waterfront" "view" "condition" "grade"   
## [13] "sqft\_above" "sqft\_basement" "yr\_built" "yr\_renovated"   
## [17] "zipcode" "lat" "long" "sqft\_living15"  
## [21] "sqft\_lot15"

naColumns <- checkNaFunction(houseData)  
if(length(naColumns)>0){  
 cat("Found NA Colums:")  
 for(i in 1:length(naColumns)) {  
 cat(sprintf("%s,", colnames(houseData)[i]))  
 }  
}  
  
#houseData$date<-(substr(houseData$date, 1, 8))  
#houseData$date<- ymd(houseData$date)  
#houseData$date<-as.numeric(as.Date(houseData$date, origin = "1900-01-01"))  
  
# Here we conclude that this data does not hold any column with NA.  
bucketByColumn(houseData,3)

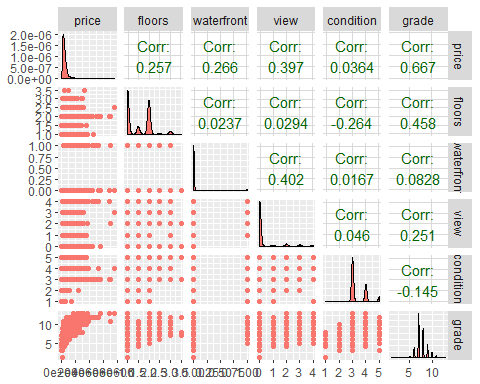
## Min-Max value for: price , MAX: 75000, MIN: 7700000

# Price with other attributes

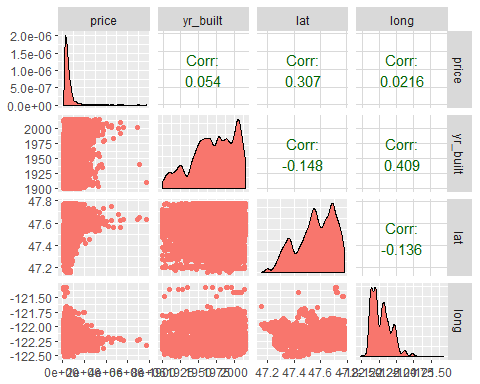
## verify the relationship between price, bedrooms, bathrooms, sqft\_living and sqft lot  
plot1 <- ggpairs(data=houseData, columns=3:7, mapping = aes(color = "dark green"), axisLabels="show")  
plot1



## 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

#Only sqft\_living & sqft\_above,sqft\_living & grade,sqft\_living & bathrooms have good correlation between them  
#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

# 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")

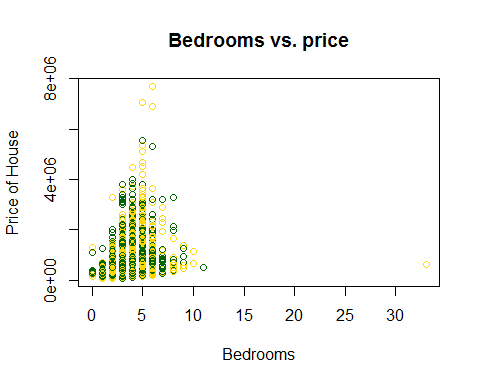
# Now Let us do analysis of price with all other variables

# Bedroom Vs Price analysis

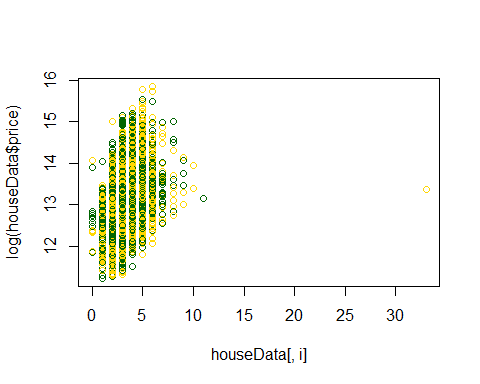
bucketByColumn(houseData,4)

## Min-Max value for: bedrooms , MAX: 0, MIN: 33

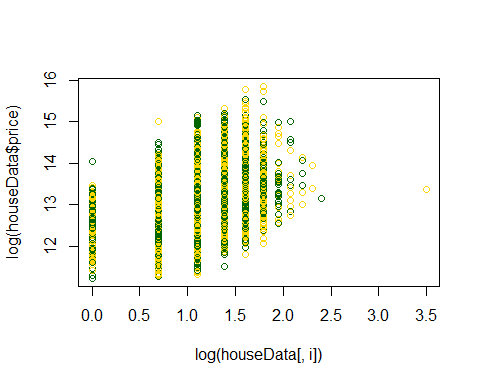
analysis(houseData,4,c('Bedrooms vs. price','Bedrooms', 'Price of House'), 'Y')



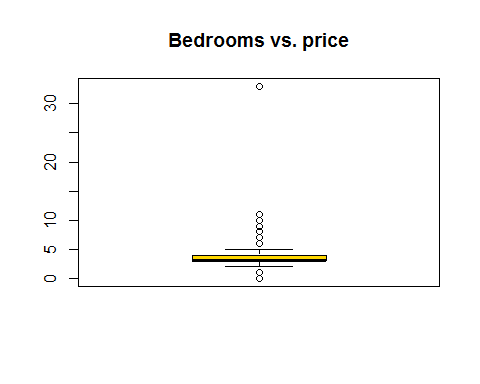
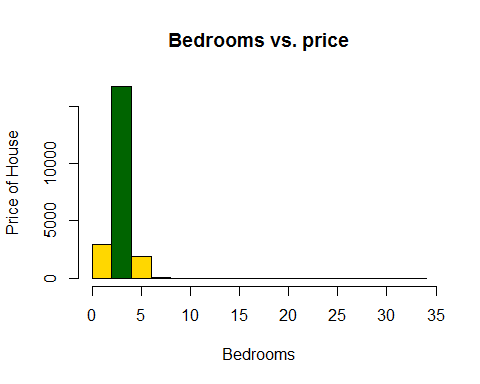
## BedroomsLog of Price of House



## Bedrooms vs. priceLog of BedroomsLog of Price of House

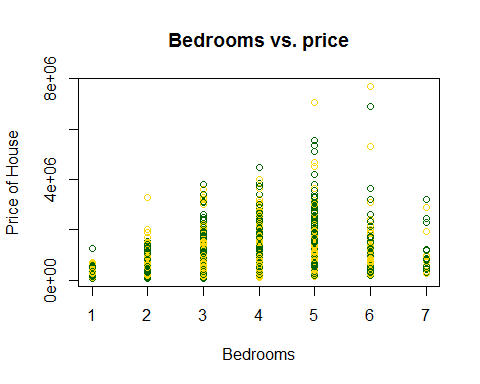


## Log Bedrooms vs. price

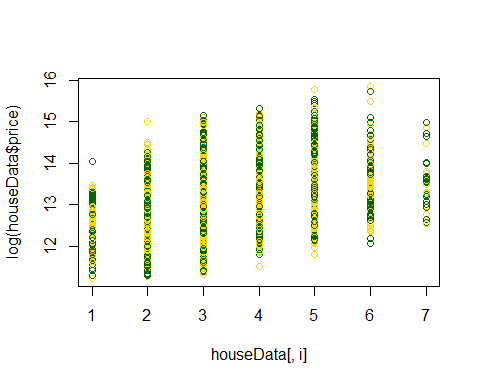


## [1] 0.3083496

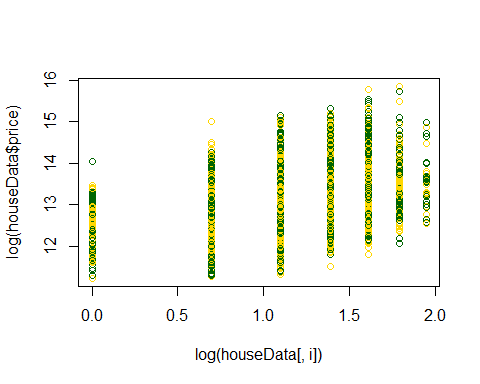
#\*\*\*\*\*\*\*Removing the outliers  
#Since more than 7 bedrooms are very rare.Also it's the outlier for my model.  
#I have removed the outlier data.  
houseData<-subset(houseData,bedrooms>=1 & bedrooms<=7)  
#\*\*\*\*\*\*\*Once we removed the outliers, again get the analysis  
analysis(houseData,4,c('Bedrooms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



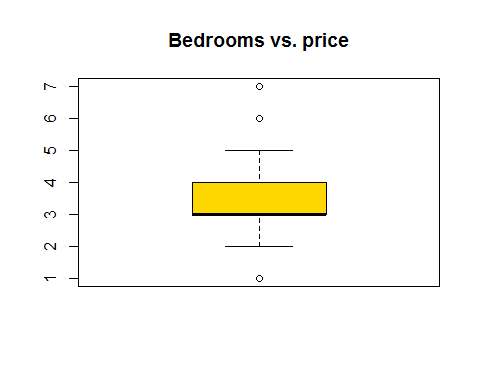
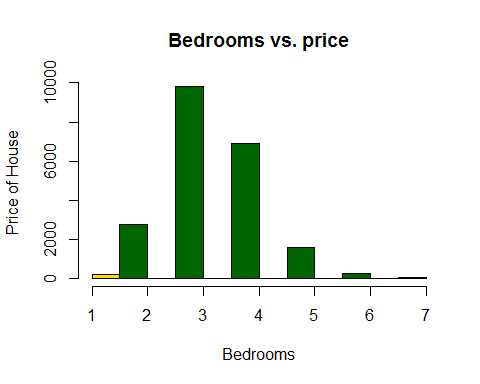
## BedroomsLog of Price of House



## Bedrooms vs. priceLog of BedroomsLog of Price of House



## Log Bedrooms vs. price



## [1] 0.3156734

bucketByColumn(houseData,4)

## Min-Max value for: bedrooms , MAX: 1, MIN: 7

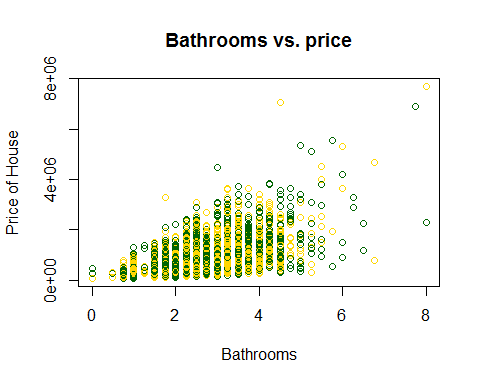
## here we found that log of bedroom give better performance.

# Bathroom Vs Price analysis

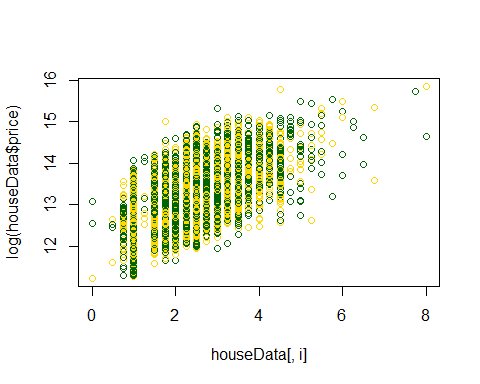
bucketByColumn(houseData,5)

## Min-Max value for: bathrooms , MAX: 0, MIN: 8

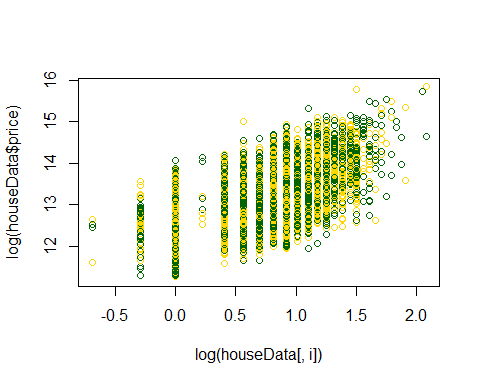
analysis(houseData,5,c('Bathrooms vs. price','Bathrooms', 'Price of House'), 'Y')



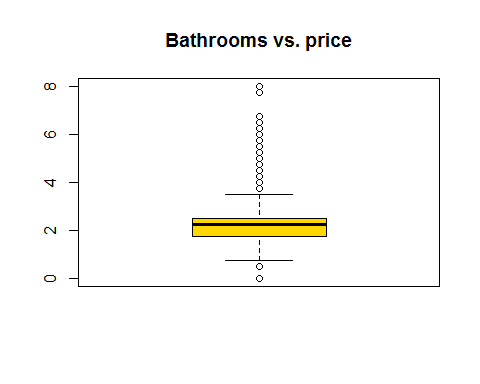
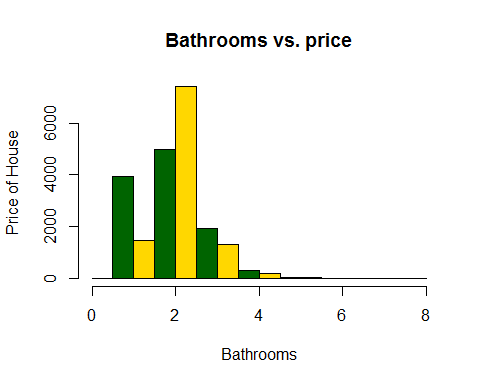
## BathroomsLog of Price of House



## Bathrooms vs. priceLog of BathroomsLog of Price of House

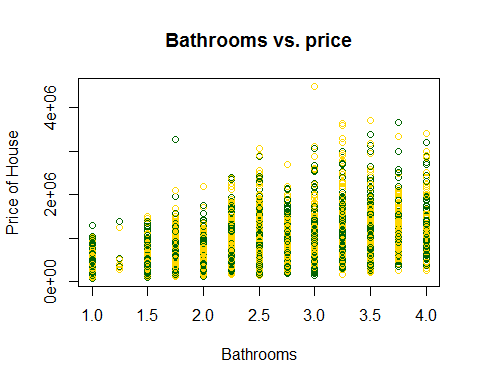


## Log Bathrooms vs. price

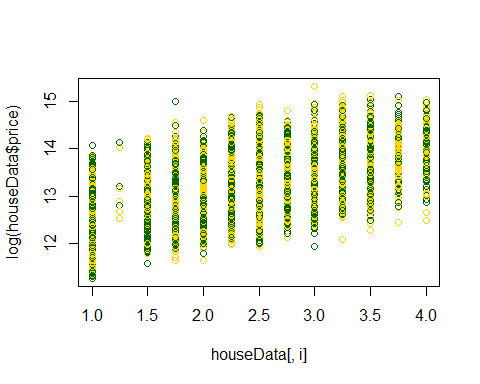


## [1] 0.5259342

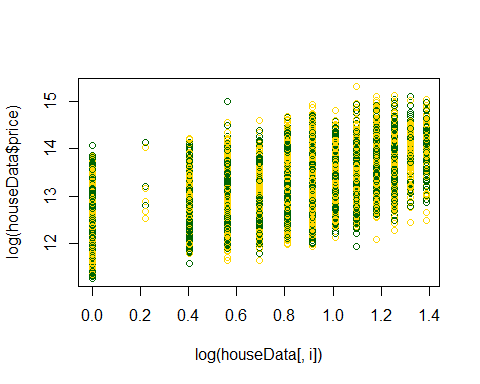
## Price vs. Bathrooms, here we can find good correlation, as number of bahtrooms increases, price increases as well, with one expection in when bathroom=7  
#\*\*\*\*\*\*\*Removing the outliers  
#More than 4 bathrooms are very rare in this data.So I am removing it.  
houseData<-subset(houseData,bathrooms>=1 & bathrooms<=4)  
analysis(houseData,5,c('Bathrooms vs. price','Bathrooms', 'Price of House'), 'Y')



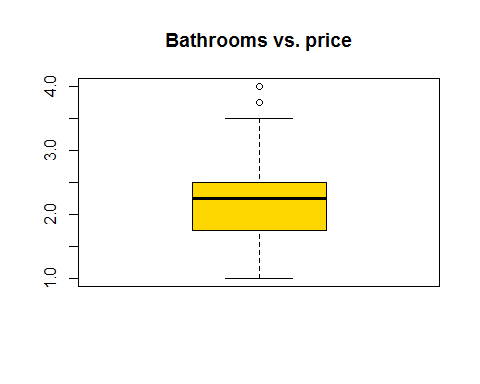
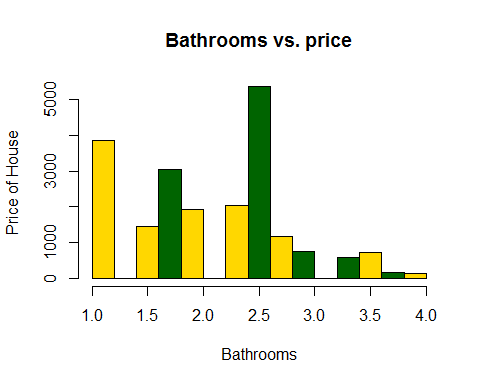
## BathroomsLog of Price of House



## Bathrooms vs. priceLog of BathroomsLog of Price of House



## Log Bathrooms vs. price



## [1] 0.475159

bucketByColumn(houseData,5)

## Min-Max value for: bathrooms , MAX: 1, MIN: 4

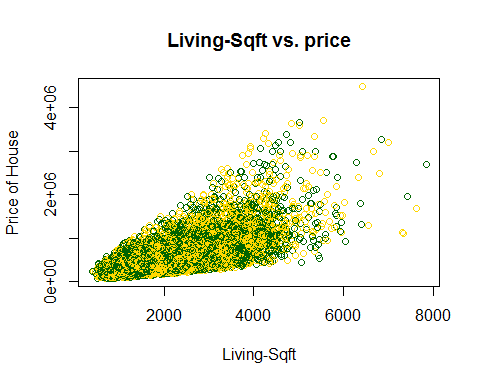
## here we found that log of bathrooms give better performance.

# SQFT Living Vs Price analysis

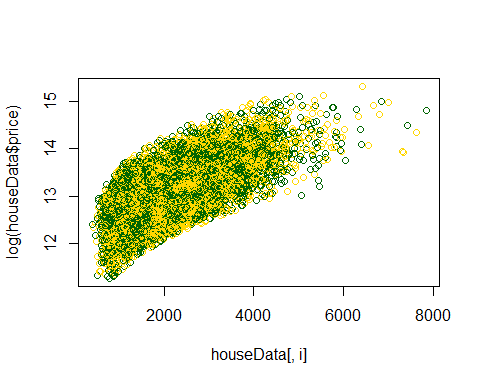
bucketByColumn(houseData,6)

## Min-Max value for: sqft\_living , MAX: 390, MIN: 7850

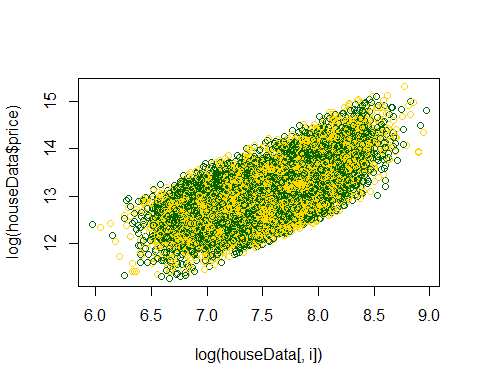
analysis(houseData,6,c('Living-Sqft vs. price','Living-Sqft', 'Price of House'), 'Y')



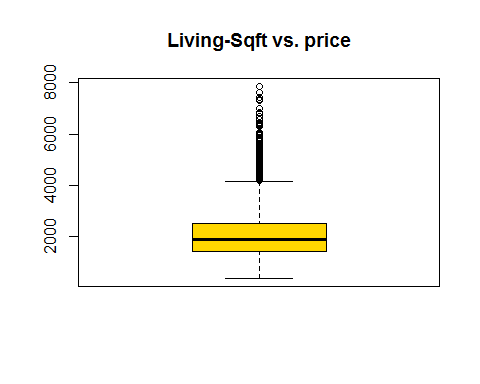
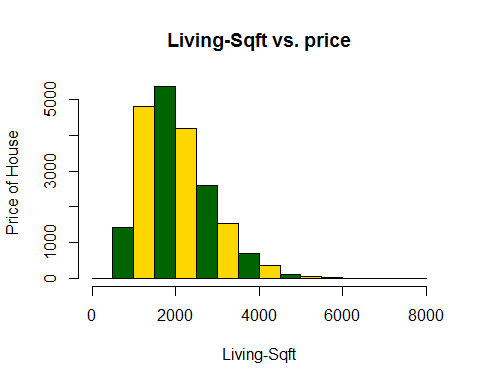
## Living-SqftLog of Price of House



## Living-Sqft vs. priceLog of Living-SqftLog of Price of House

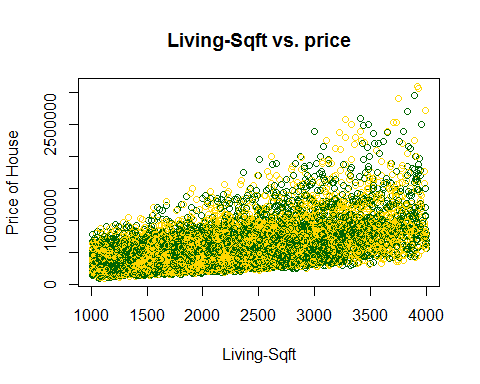


## Log Living-Sqft vs. price

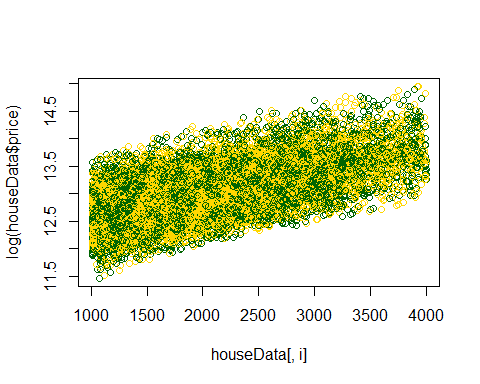


## [1] 0.6701029

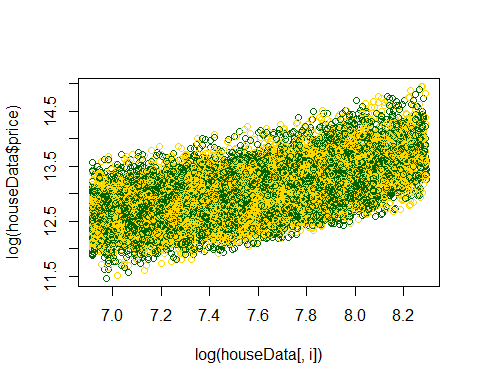
## Price vs. Sqft\_living ->> Nice correlation, as sqft increases, price increases as well.  
#\*\*\*\*\*\*\*Removing the outliers  
houseData<-subset(houseData,sqft\_living >1000 & sqft\_living<=4000)  
analysis(houseData,6,c('Living-Sqft vs. price','Living-Sqft', 'Price of House'), 'Y')



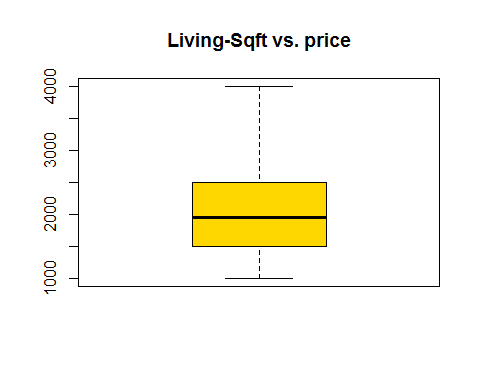
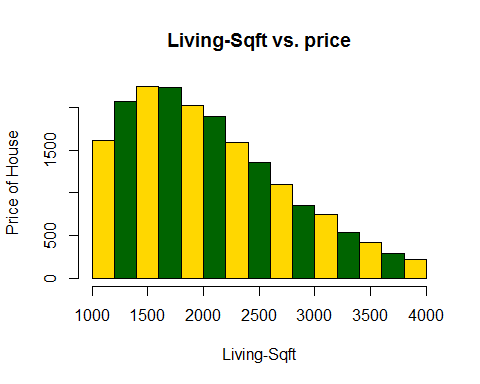
## Living-SqftLog of Price of House



## Living-Sqft vs. priceLog of Living-SqftLog of Price of House



## Log Living-Sqft vs. price



## [1] 0.5938015

bucketByColumn(houseData,6)

## Min-Max value for: sqft\_living , MAX: 1008, MIN: 4000

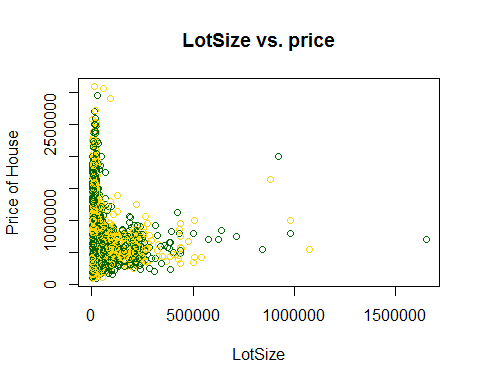
## Price vs. Sqft\_living ->> Nice correlation, as sqft increases, price increases as well.

## SQFT\_LOT Vs Price analysis

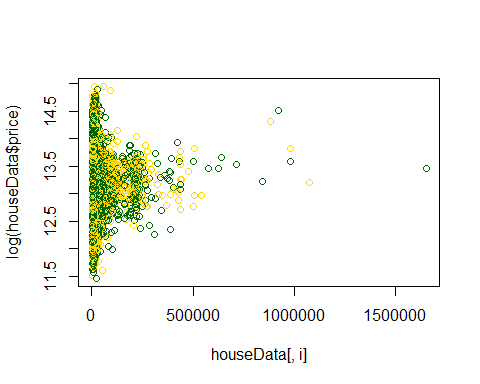
bucketByColumn(houseData,7)

## Min-Max value for: sqft\_lot , MAX: 520, MIN: 1651359

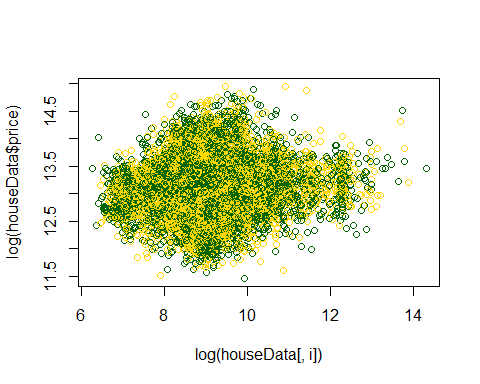
analysis(houseData,7,c('LotSize vs. price','LotSize', 'Price of House'), 'Y')



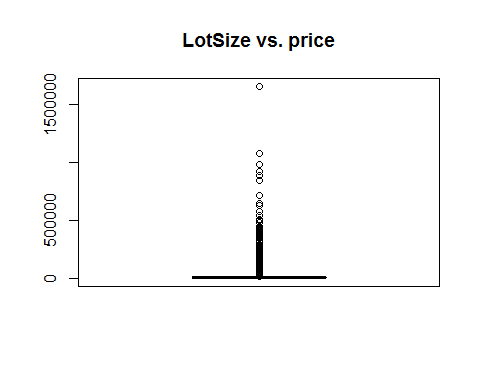
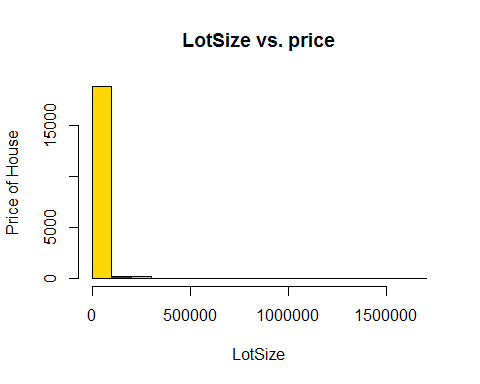
## LotSizeLog of Price of House



## LotSize vs. priceLog of LotSizeLog of Price of House

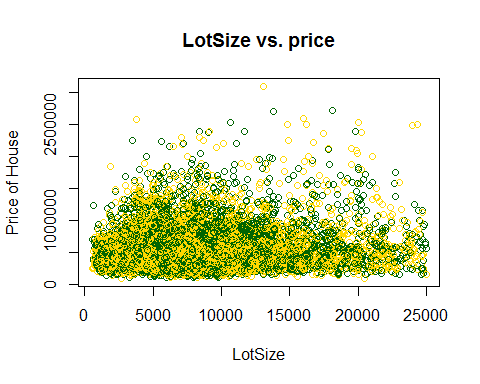


## Log LotSize vs. price

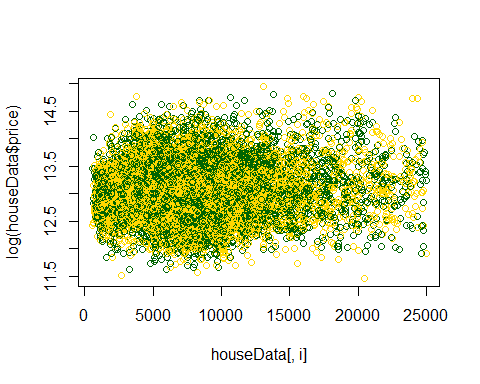


## [1] 0.06105332

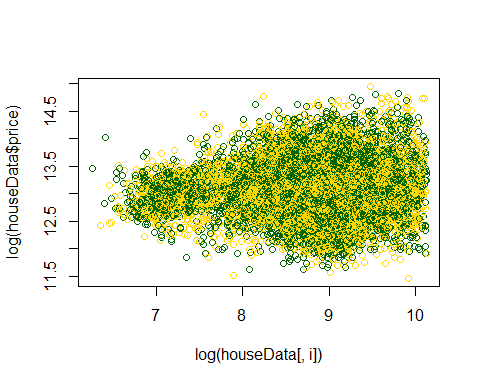
#\*\*\*\*\*\*\*Removing the outliers  
houseData<-subset(houseData,sqft\_lot>=0 & sqft\_lot<=25000)  
analysis(houseData,7,c('LotSize vs. price','LotSize', 'Price of House'), 'Y')



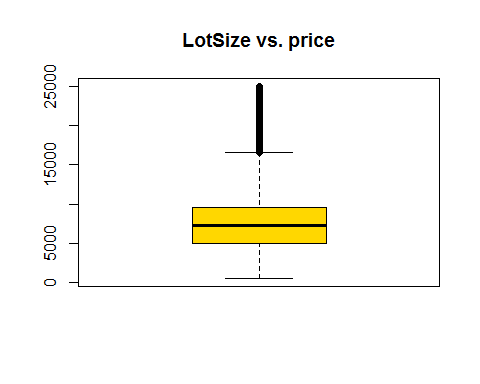
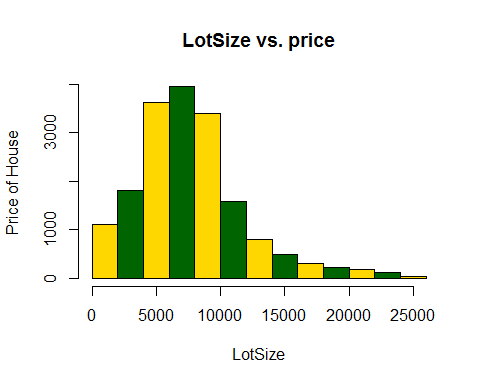
## LotSizeLog of Price of House



## LotSize vs. priceLog of LotSizeLog of Price of House



## Log LotSize vs. price



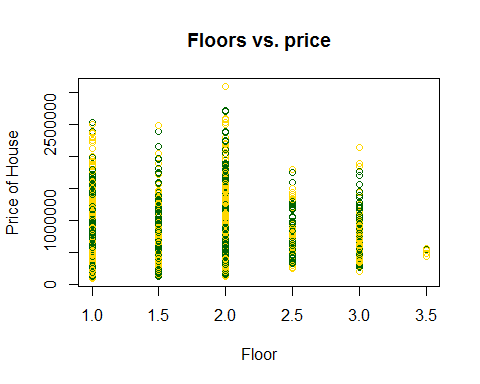
## [1] 0.06714415

bucketByColumn(houseData,7)

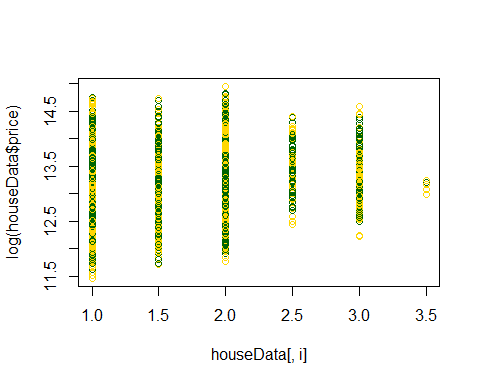
## Min-Max value for: sqft\_lot , MAX: 520, MIN: 25000

## FLOOR Vs Price analysis

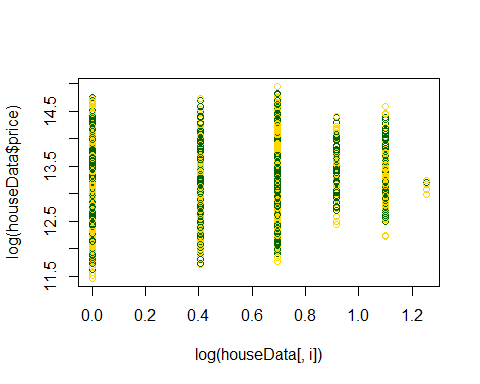
analysis(houseData,8,c('Floors vs. price','Floor', 'Price of House'), 'Y')



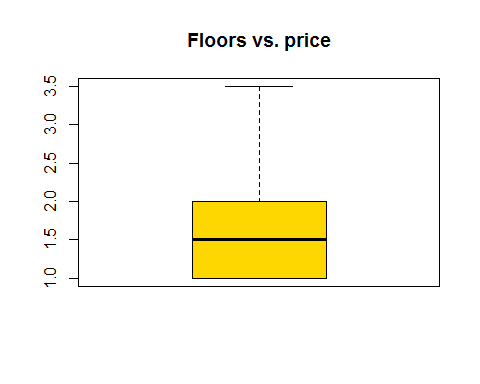
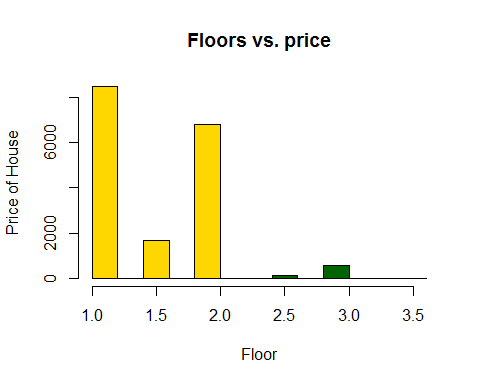
## FloorLog of Price of House



## Floors vs. priceLog of FloorLog of Price of House



## Log Floors vs. price



## [1] 0.2072373

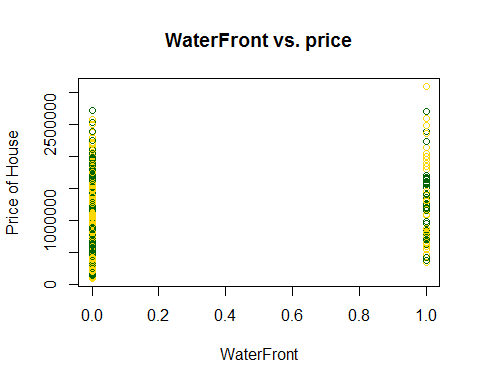
#bucketByColumn(houseData,8)

## SQFT\_LOT Vs Price analysis

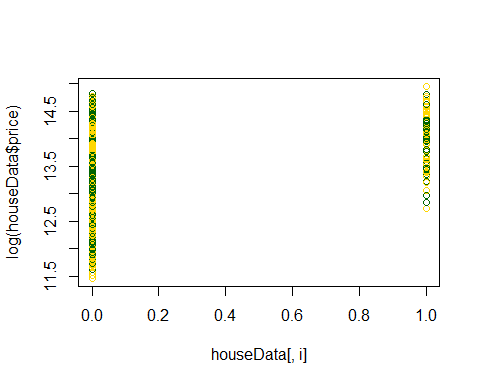
bucketByColumn(houseData,9)

## Min-Max value for: waterfront , MAX: 0, MIN: 1

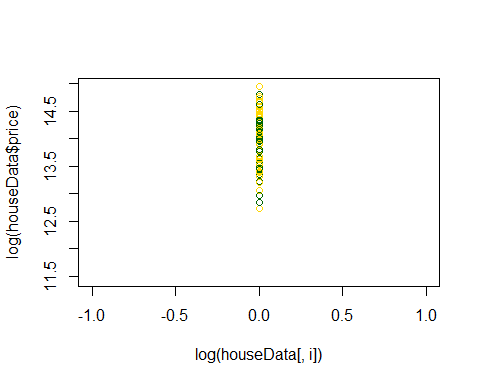
analysis(houseData,9,c('WaterFront vs. price','WaterFront', 'Price of House'), 'Y')



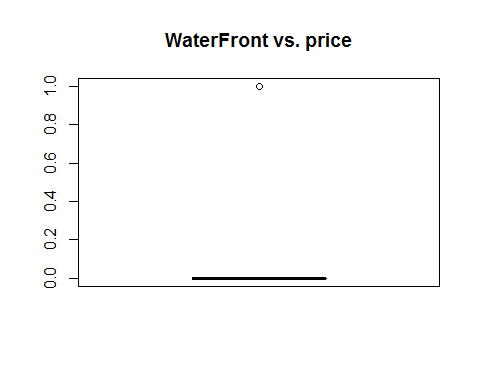
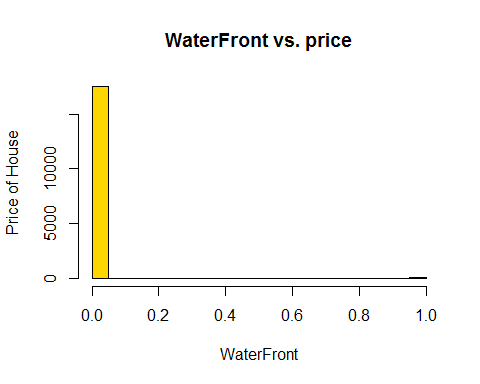
## WaterFrontLog of Price of House



## WaterFront vs. priceLog of WaterFrontLog of Price of House

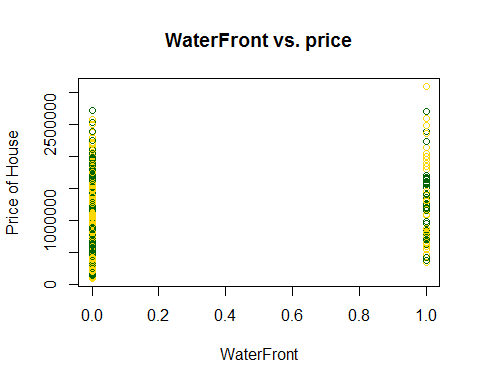


## Log WaterFront vs. price

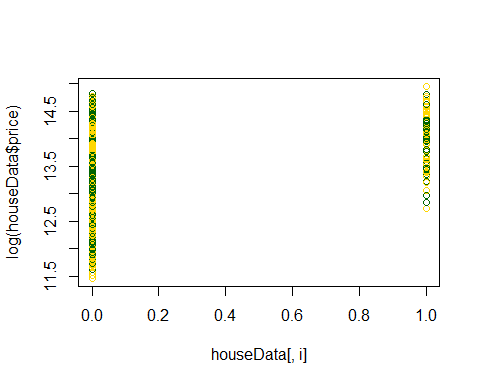


## [1] 0.209102

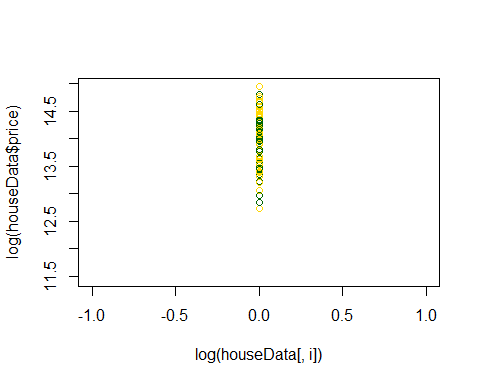
#\*\*\*\*\*\*\*Removing the outliers  
houseData<-subset(houseData,sqft\_lot>=0 & sqft\_lot<=25000)  
analysis(houseData,9,c('WaterFront vs. price','WaterFront', 'Price of House'), 'Y')



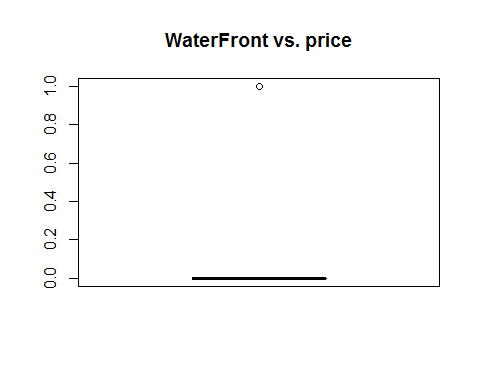
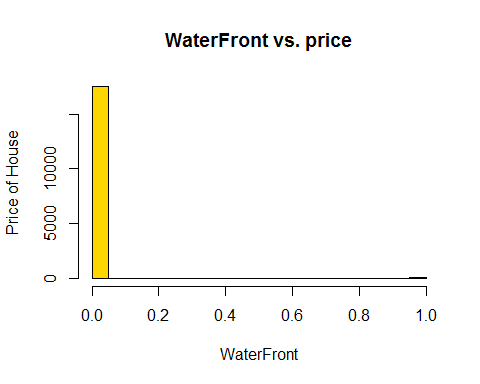
## WaterFrontLog of Price of House



## WaterFront vs. priceLog of WaterFrontLog of Price of House



## Log WaterFront vs. price



## [1] 0.209102

bucketByColumn(houseData,9)

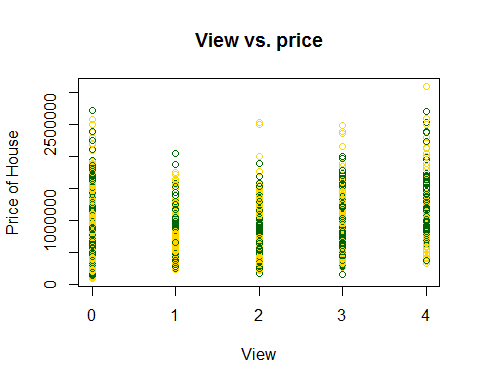
## Min-Max value for: waterfront , MAX: 0, MIN: 1

## View Vs Price analysis

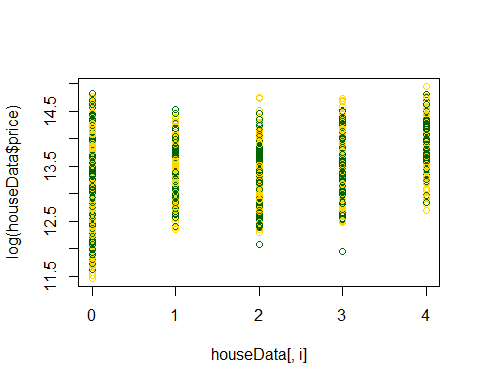
bucketByColumn(houseData,10)

## Min-Max value for: view , MAX: 0, MIN: 4

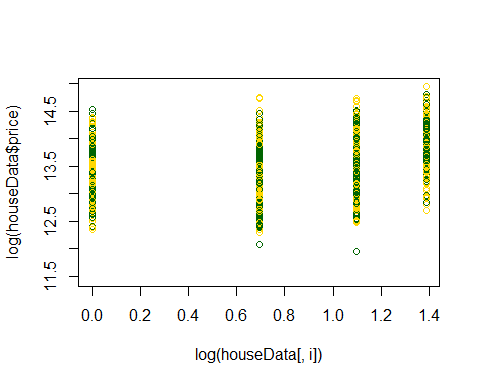
analysis(houseData,10,c('View vs. price','View', 'Price of House'), 'Y')



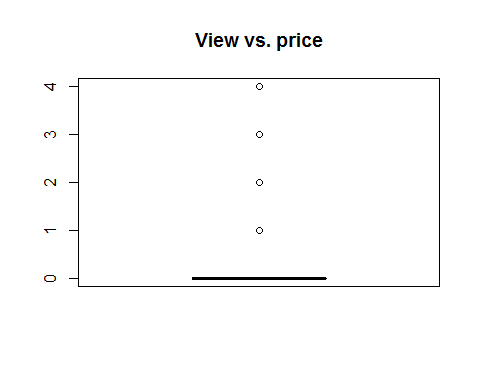
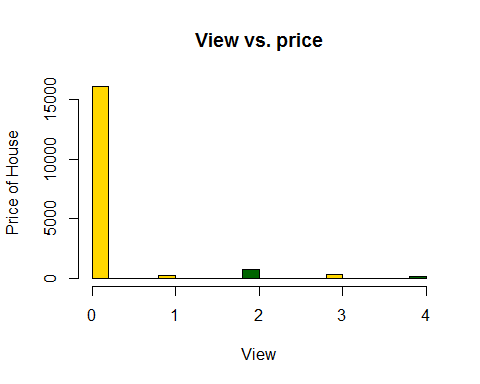
## ViewLog of Price of House



## View vs. priceLog of ViewLog of Price of House

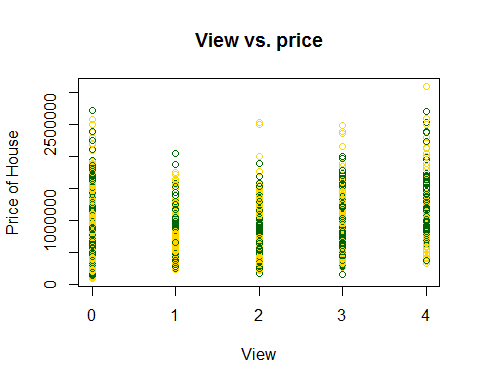


## Log View vs. price

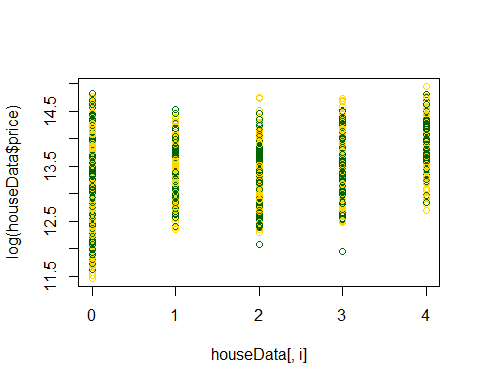


## [1] 0.3578146

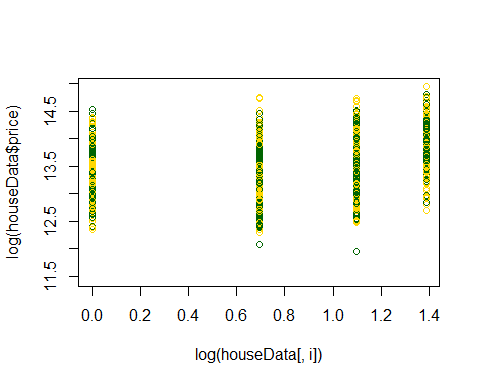
## Price vs. View ->> Nice correlation, view increases [median of bar plot], price increases as well  
#\*\*\*\*\*\*\*Removing the outliers  
analysis(houseData,10,c('View vs. price','View', 'Price of House'), 'Y')



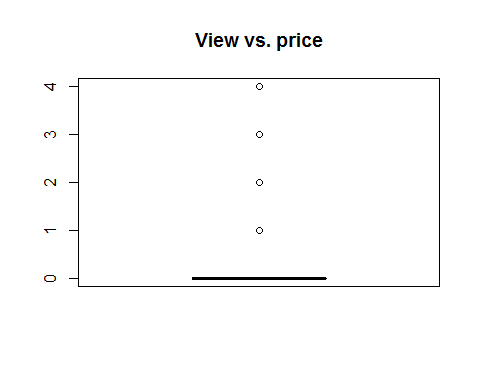
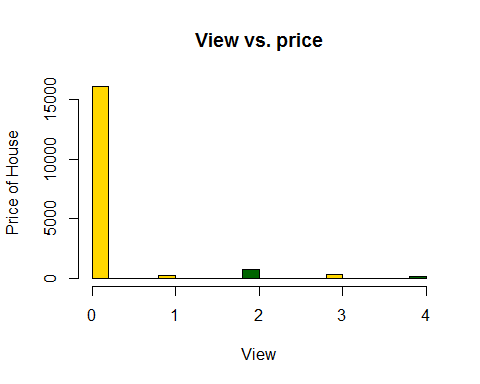
## ViewLog of Price of House



## View vs. priceLog of ViewLog of Price of House



## Log View vs. price



## [1] 0.3578146

bucketByColumn(houseData,10)

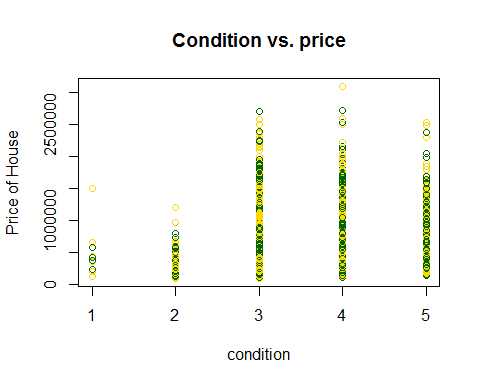
## Min-Max value for: view , MAX: 0, MIN: 4

## CONDITION Vs Price analysis

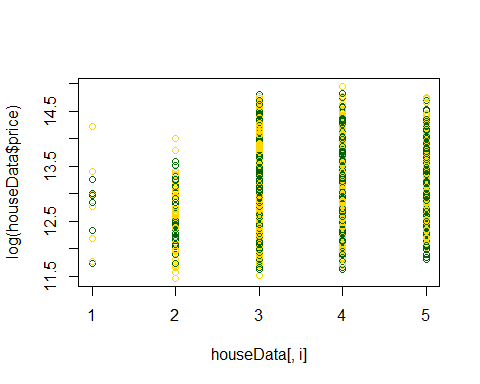
bucketByColumn(houseData,11)

## Min-Max value for: condition , MAX: 1, MIN: 5

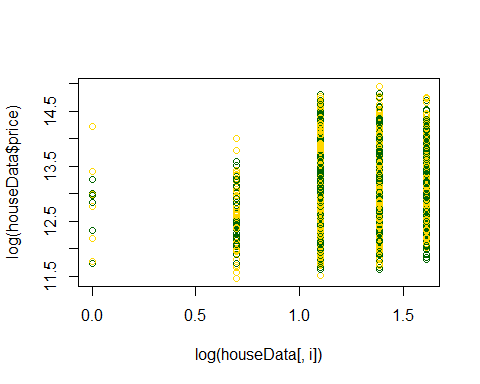
analysis(houseData,11,c('Condition vs. price','condition', 'Price of House'), 'Y')



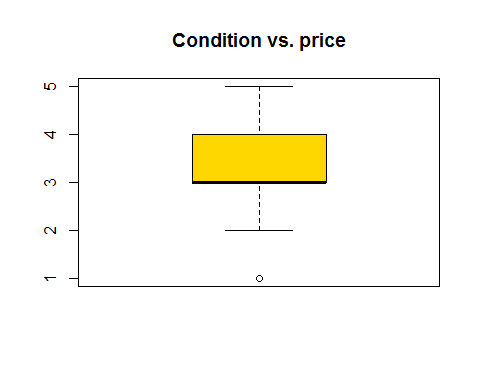
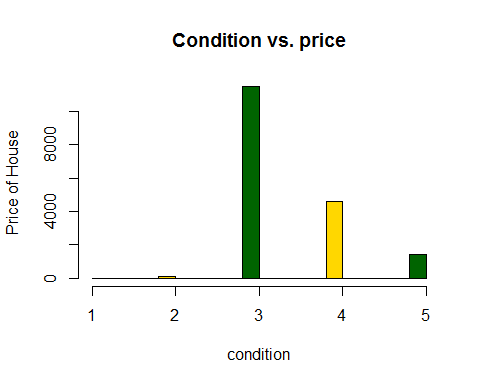
## conditionLog of Price of House



## Condition vs. priceLog of conditionLog of Price of House

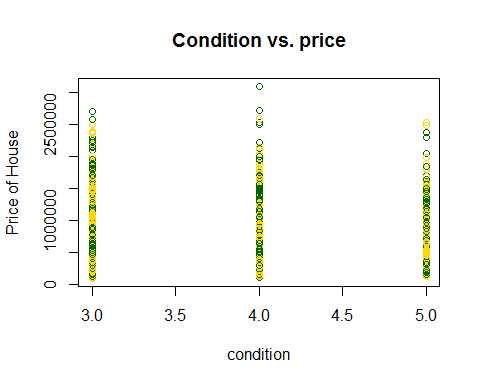


## Log Condition vs. price

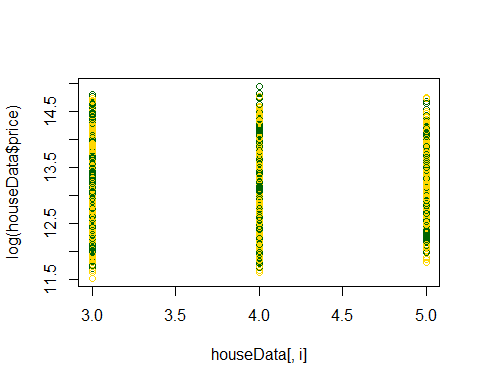


## [1] 0.07301379

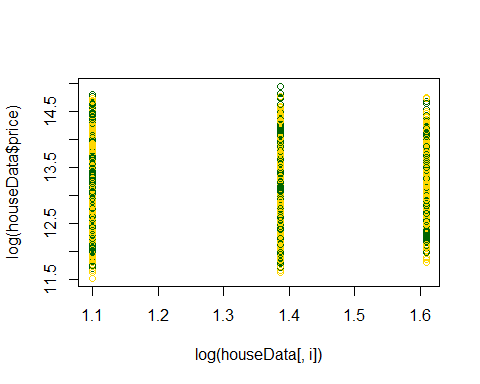
#\*\*\*\*\*\*\*Removing the outliers  
houseData<-subset(houseData,condition>=3& condition<=5)  
analysis(houseData,11,c('Condition vs. price','condition', 'Price of House'), 'Y')



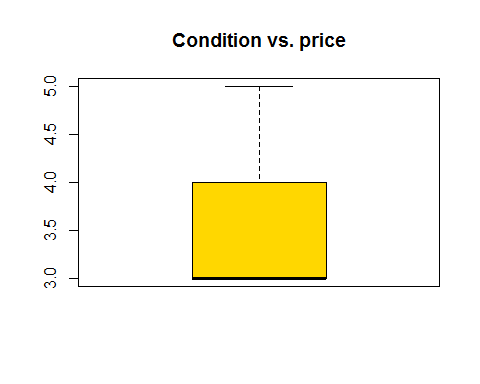
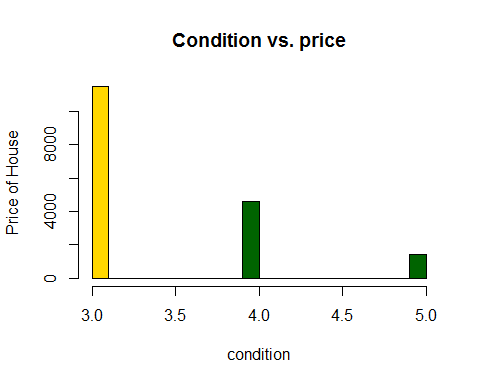
## conditionLog of Price of House



## Condition vs. priceLog of conditionLog of Price of House



## Log Condition vs. price



## [1] 0.06582398

bucketByColumn(houseData,11)

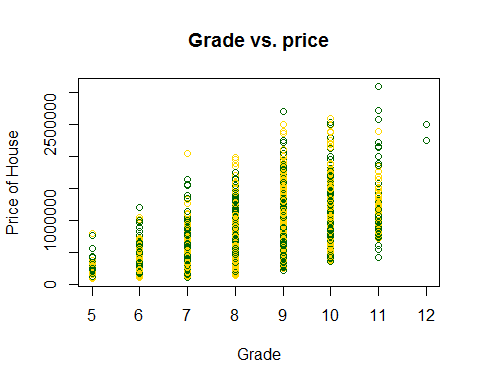
## Min-Max value for: condition , MAX: 3, MIN: 5

## Grade Vs Price analysis

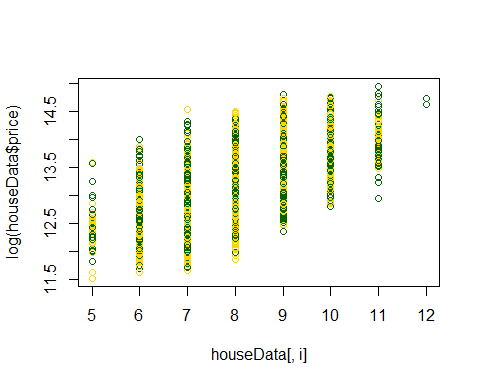
bucketByColumn(houseData,12)

## Min-Max value for: grade , MAX: 5, MIN: 12

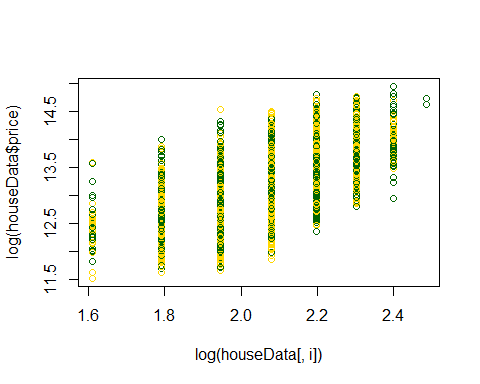
analysis(houseData,12,c('Grade vs. price','Grade', 'Price of House'), 'Y')



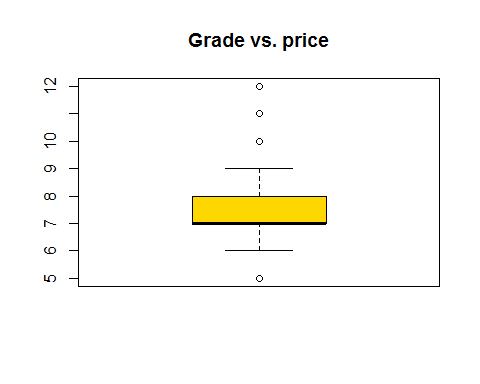
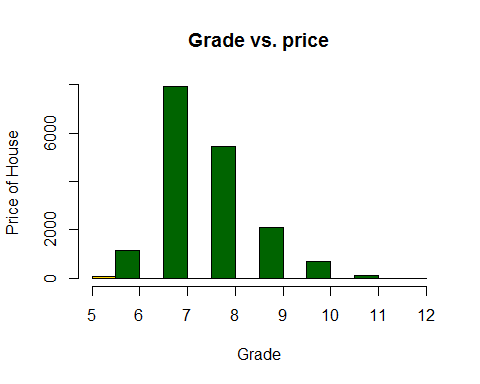
## GradeLog of Price of House



## Grade vs. priceLog of GradeLog of Price of House

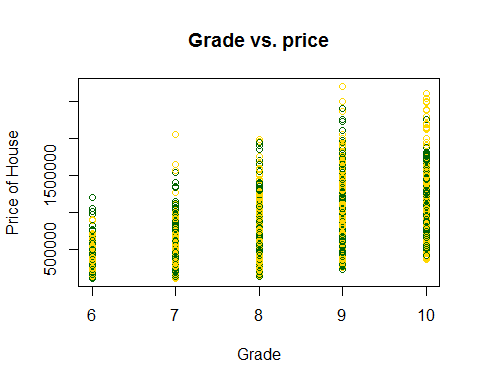


## Log Grade vs. price

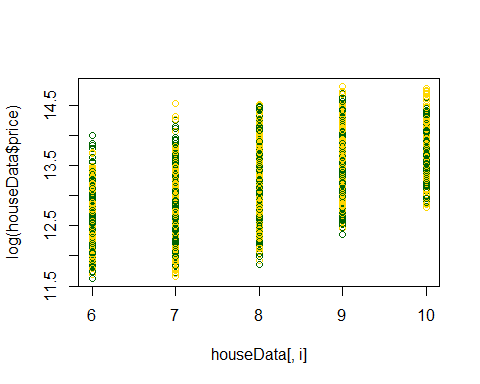


## [1] 0.6086724

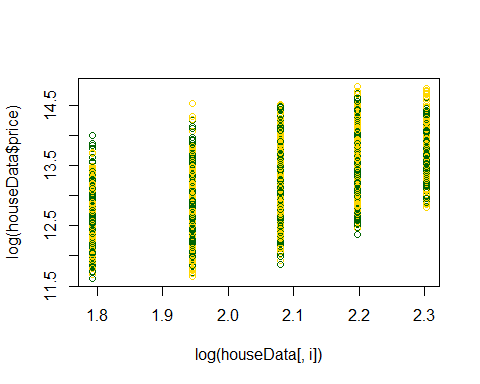
## Price vs. Grade ->> Nice correlation, grade increases [median of bar plot], price increases as well  
#\*\*\*\*\*\*\*Removing the outliers  
#Most of the houses grades are between 6-10   
houseData<-subset(houseData,grade >= 6 & grade<=10)  
analysis(houseData,12,c('Grade vs. price','Grade', 'Price of House'), 'Y')



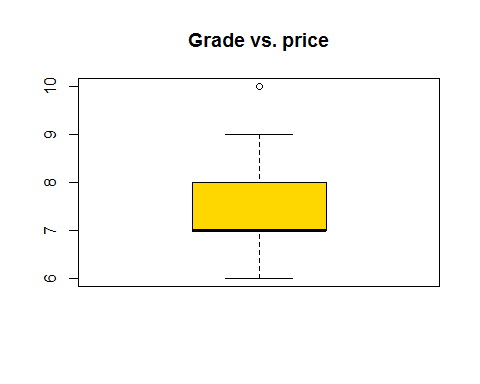
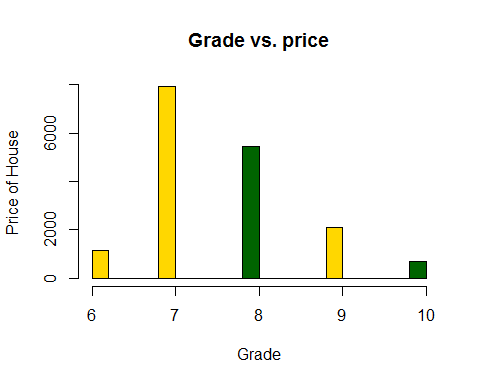
## GradeLog of Price of House



## Grade vs. priceLog of GradeLog of Price of House



## Log Grade vs. price



## [1] 0.5926375

bucketByColumn(houseData,12)

## Min-Max value for: grade , MAX: 6, MIN: 10

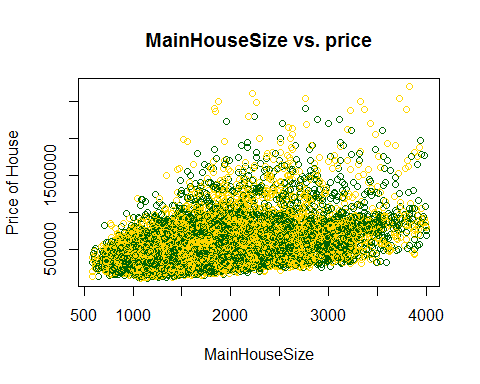
#grade is good without log

# SQFT\_ABOVE Vs Price analysis

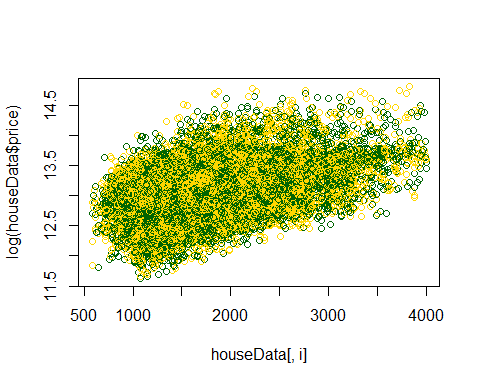
bucketByColumn(houseData,13)

## Min-Max value for: sqft\_above , MAX: 580, MIN: 4000

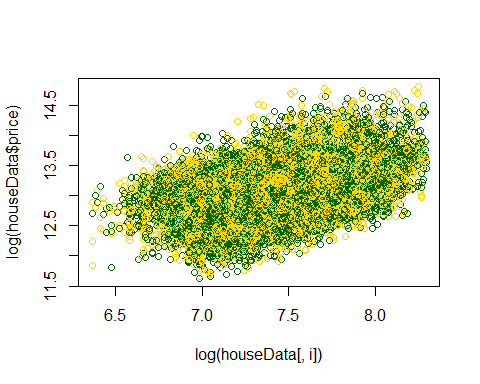
analysis(houseData,13,c('MainHouseSize vs. price','MainHouseSize', 'Price of House'), 'Y')



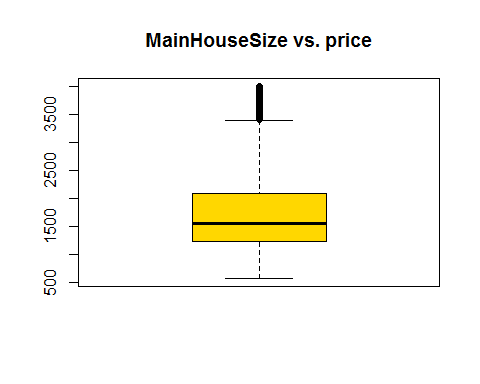
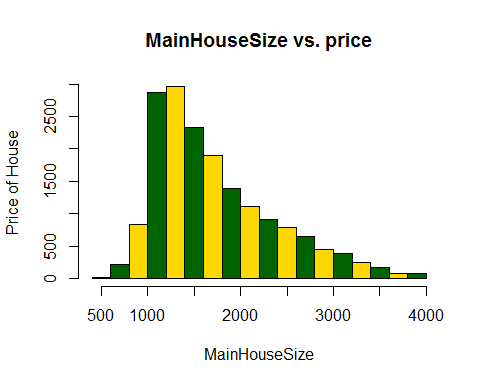
## MainHouseSizeLog of Price of House



## MainHouseSize vs. priceLog of MainHouseSizeLog of Price of House

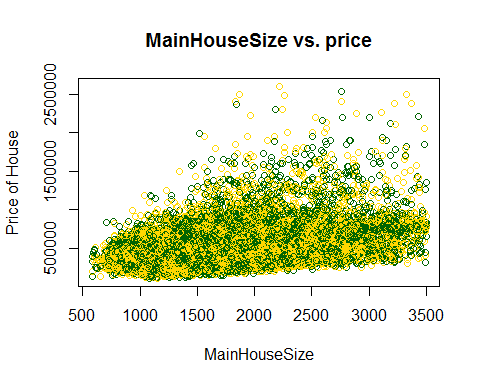


## Log MainHouseSize vs. price

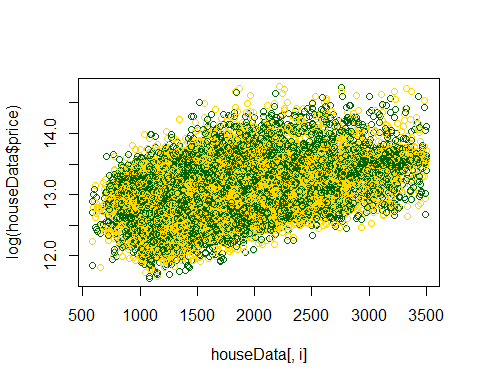


## [1] 0.4454655

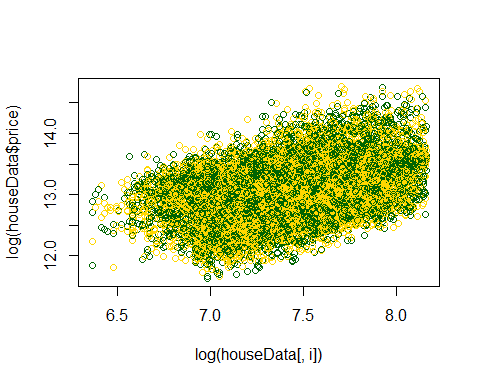
#\*\*\*\*\*\*\*Removing the outliers  
houseData<-subset(houseData,sqft\_above >=500 & sqft\_above<=3500)  
analysis(houseData,13,c('MainHouseSize vs. price','MainHouseSize', 'Price of House'), 'Y')



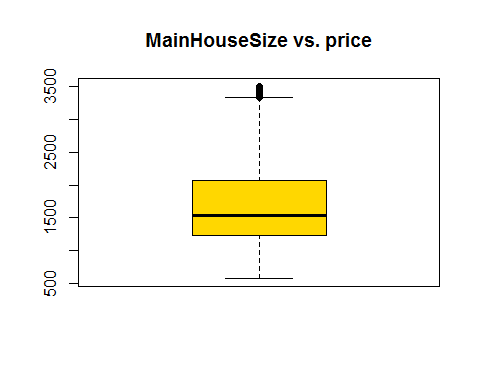
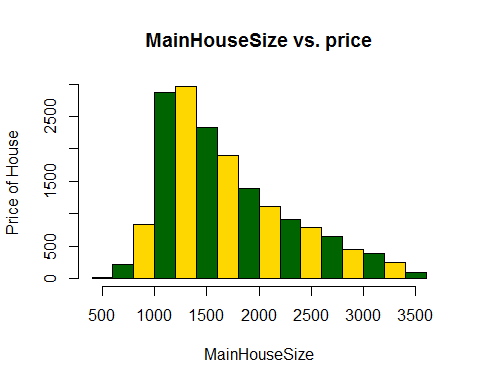
## MainHouseSizeLog of Price of House



## MainHouseSize vs. priceLog of MainHouseSizeLog of Price of House



## Log MainHouseSize vs. price



## [1] 0.4176138

bucketByColumn(houseData,13)

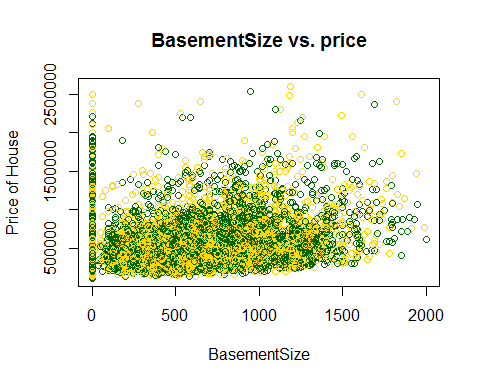
## Min-Max value for: sqft\_above , MAX: 580, MIN: 3500

## SQFT\_BASEMENT Vs Price analysis

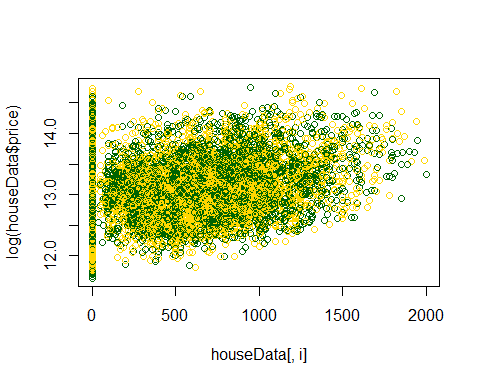
bucketByColumn(houseData,14)

## Min-Max value for: sqft\_basement , MAX: 0, MIN: 2000

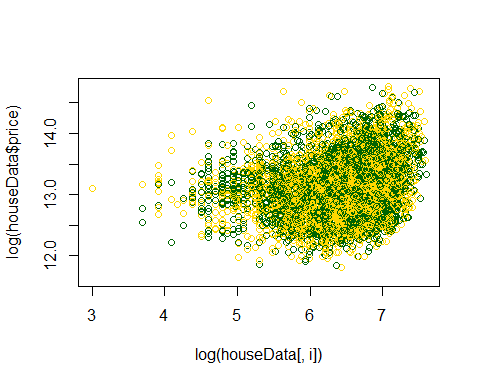
analysis(houseData,14,c('BasementSize vs. price','BasementSize', 'Price of House'), 'Y')



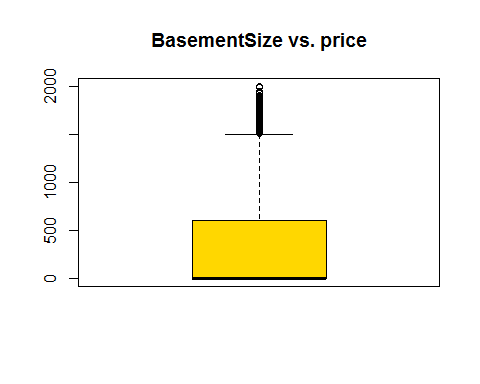
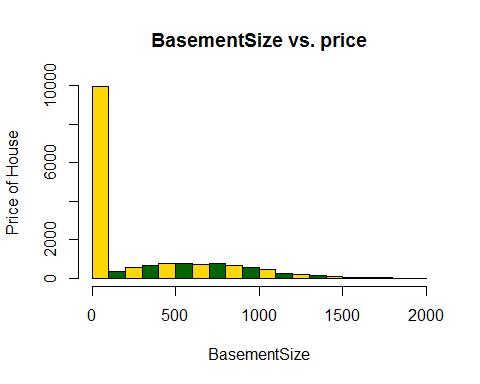
## BasementSizeLog of Price of House



## BasementSize vs. priceLog of BasementSizeLog of Price of House

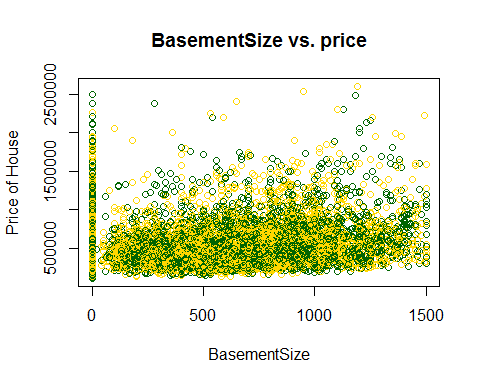


## Log BasementSize vs. price

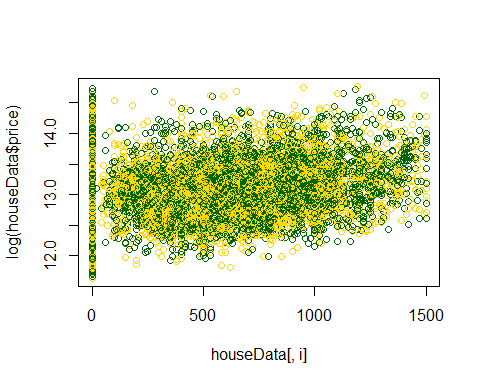


## [1] 0.2686956

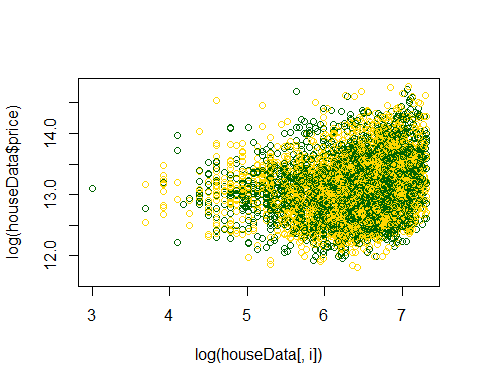
#\*\*\*\*\*\*\*Removing the outliers  
houseData<-subset(houseData,sqft\_basement >=0 & sqft\_basement<=1500)  
analysis(houseData,14,c('BasementSize vs. price','BasementSize', 'Price of House'), 'Y')



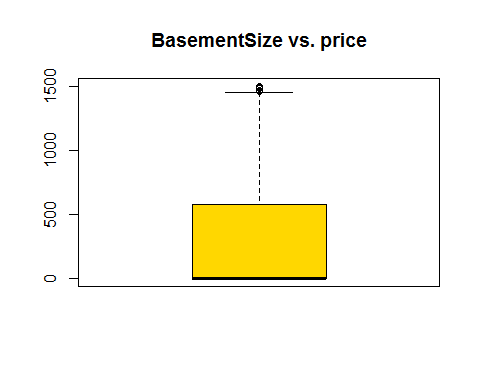
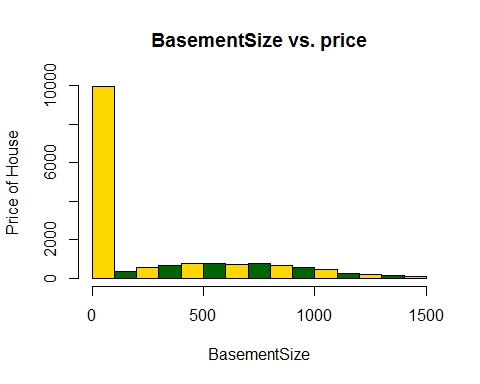
## BasementSizeLog of Price of House



## BasementSize vs. priceLog of BasementSizeLog of Price of House



## Log BasementSize vs. price



## [1] 0.2404668

bucketByColumn(houseData,14)

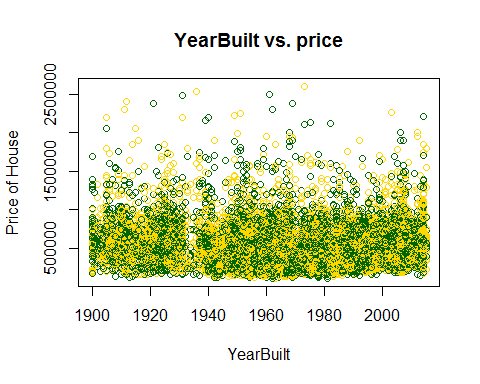
## Min-Max value for: sqft\_basement , MAX: 0, MIN: 1500

# YR\_BUILT Vs Price analysis

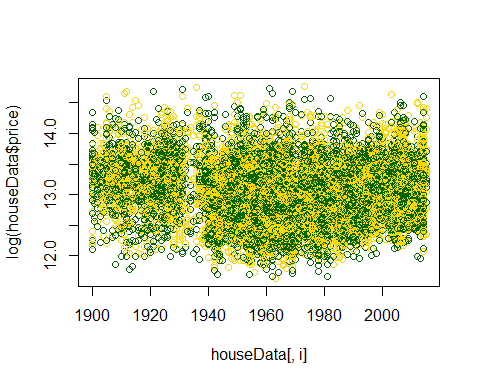
bucketByColumn(houseData,15)

## Min-Max value for: yr\_built , MAX: 1900, MIN: 2015

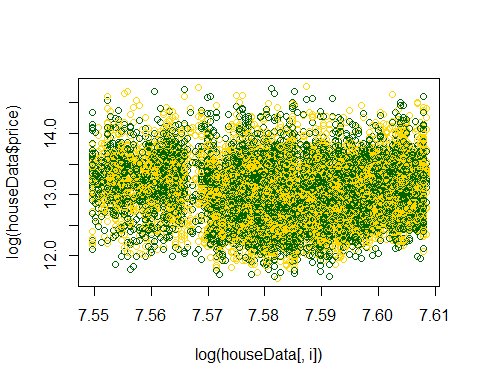
analysis(houseData,15,c('YearBuilt vs. price','YearBuilt', 'Price of House'), 'Y')



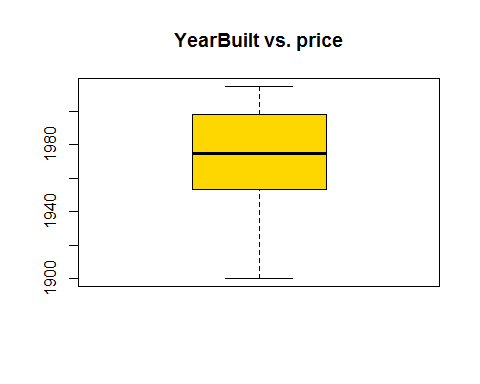
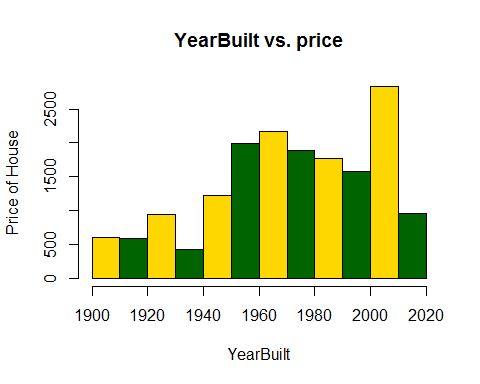
## YearBuiltLog of Price of House



## YearBuilt vs. priceLog of YearBuiltLog of Price of House

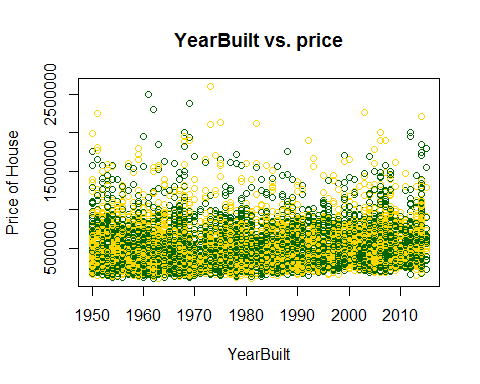


## Log YearBuilt vs. price

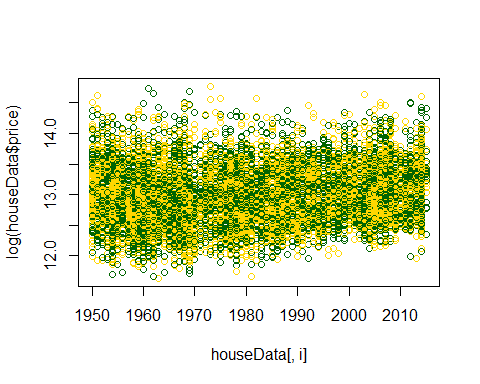


## [1] -0.09523834

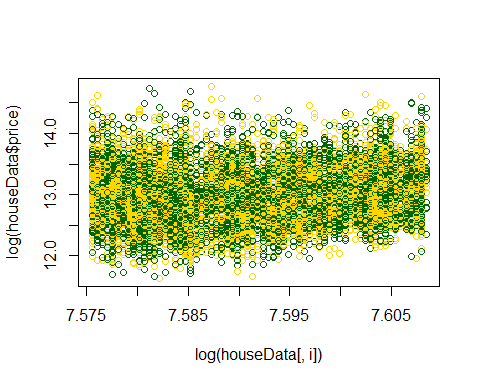
#\*\*\*\*\*\*\*Removing the outliers  
#In our data some records are too old..I just removed that data from my model.  
#Because It doesn't make any sense to keep more than 100 years house in our model  
houseData<-subset(houseData,yr\_built>=1950& yr\_built<=2015)  
analysis(houseData,15,c('YearBuilt vs. price','YearBuilt', 'Price of House'), 'Y')



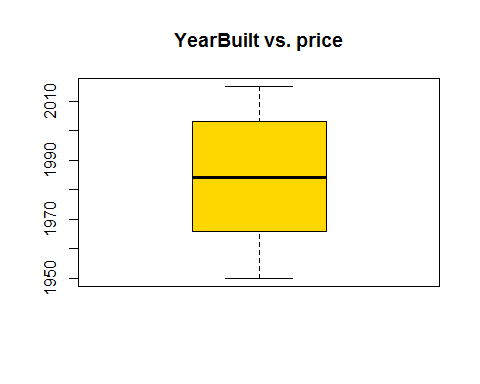
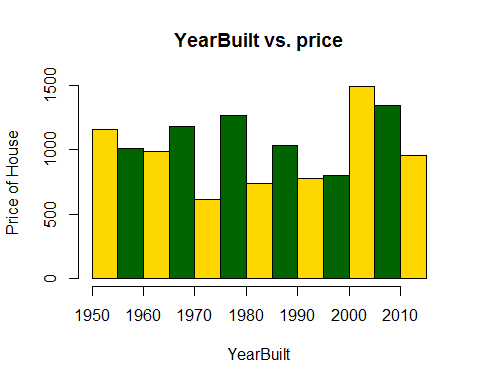
## YearBuiltLog of Price of House



## YearBuilt vs. priceLog of YearBuiltLog of Price of House



## Log YearBuilt vs. price



## [1] 0.100292

bucketByColumn(houseData,15)

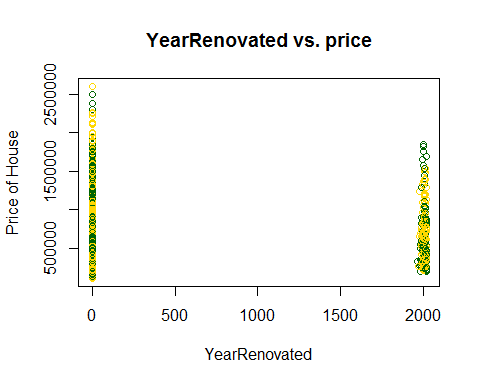
## Min-Max value for: yr\_built , MAX: 1950, MIN: 2015

# YR\_BUILT Vs Price analysis

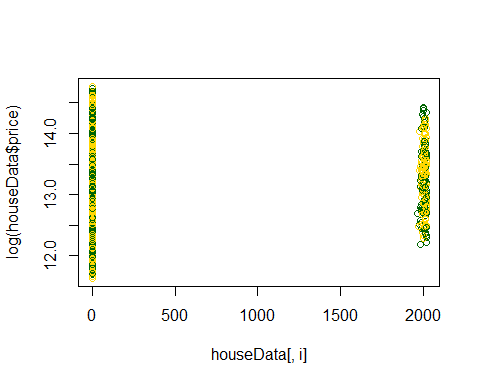
bucketByColumn(houseData,16)

## Min-Max value for: yr\_renovated , MAX: 0, MIN: 2015

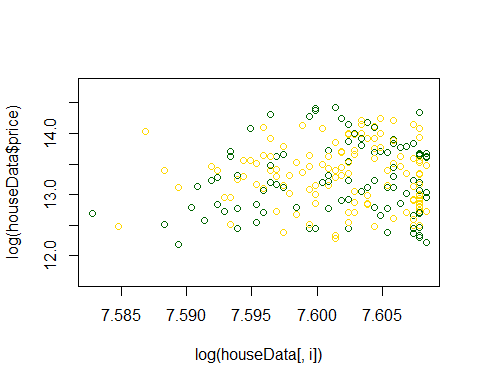
analysis(houseData,16,c('YearRenovated vs. price','YearRenovated', 'Price of House'), 'Y')



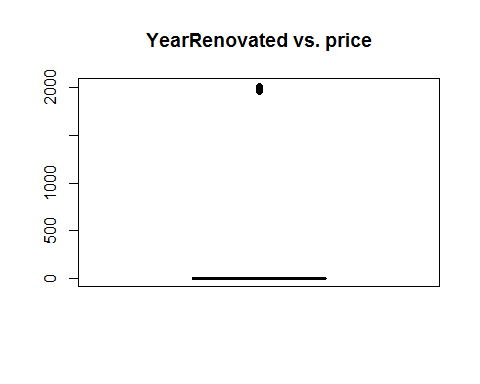
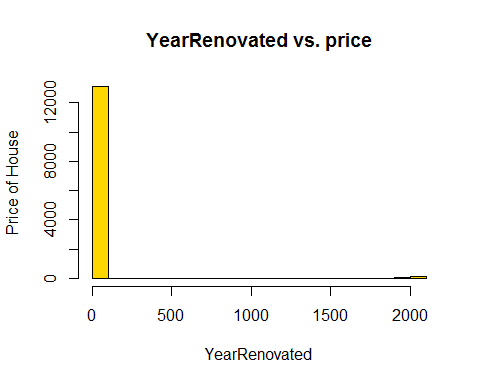
## YearRenovatedLog of Price of House



## YearRenovated vs. priceLog of YearRenovatedLog of Price of House

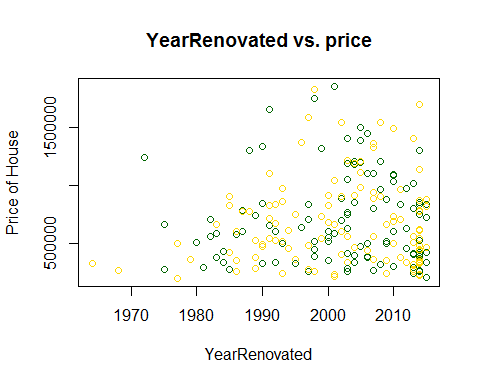


## Log YearRenovated vs. price

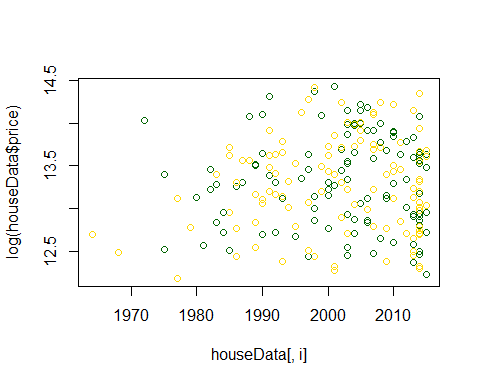


## [1] 0.1115179

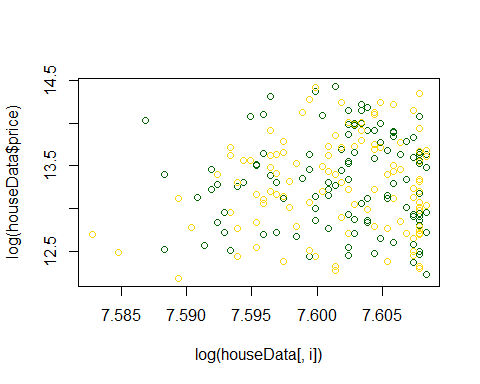
#\*\*\*\*\*\*\*Removing the outliers  
#In our data some records are too old..I just removed that data from my model.  
#Because It doesn't make any sense to keep more than 100 years house in our model  
houseData<-subset(houseData,yr\_renovated>=1950& yr\_renovated<=2015)  
analysis(houseData,16,c('YearRenovated vs. price','YearRenovated', 'Price of House'), 'Y')



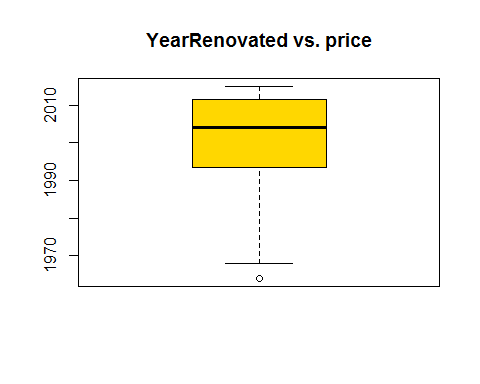
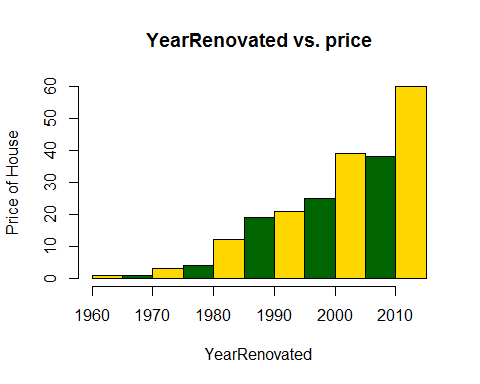
## YearRenovatedLog of Price of House



## YearRenovated vs. priceLog of YearRenovatedLog of Price of House



## Log YearRenovated vs. price



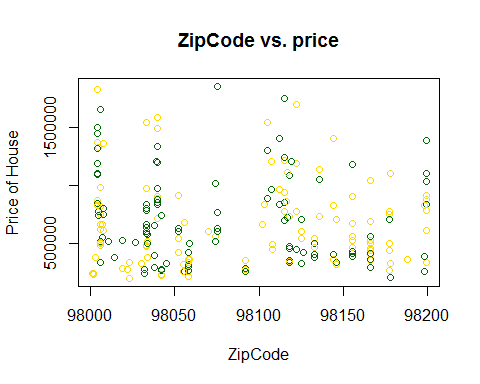
## [1] 0.05704174

bucketByColumn(houseData,16)

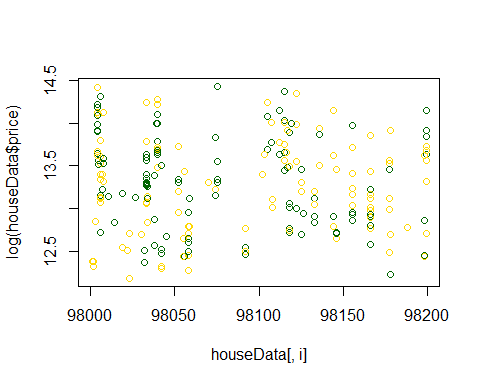
## Min-Max value for: yr\_renovated , MAX: 1964, MIN: 2015

# ZIPCODE Vs Price analysis

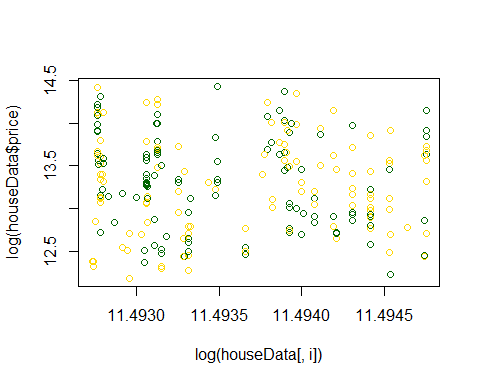
analysis(houseData,17,c('ZipCode vs. price','ZipCode', 'Price of House'), 'Y')



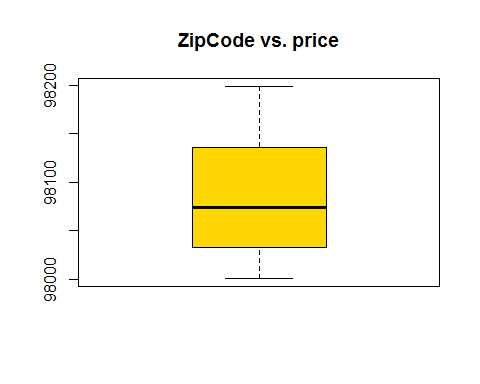
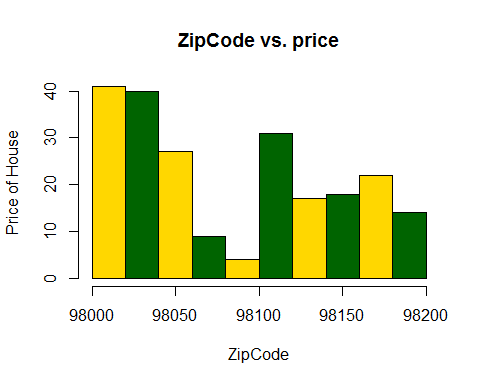
## ZipCodeLog of Price of House



## ZipCode vs. priceLog of ZipCodeLog of Price of House



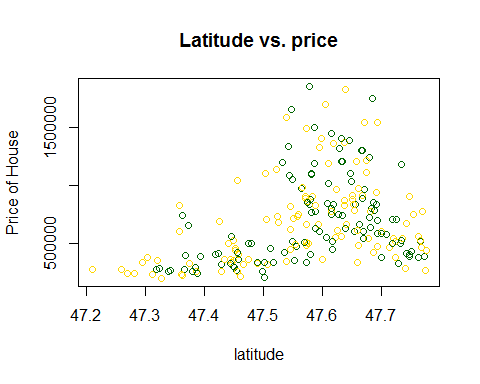
## Log ZipCode vs. price



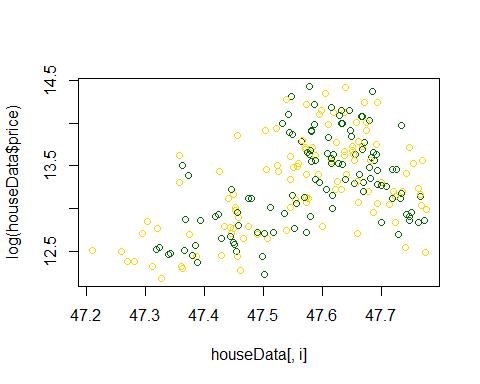
## [1] -0.07696891

## LAT Vs Price analysis

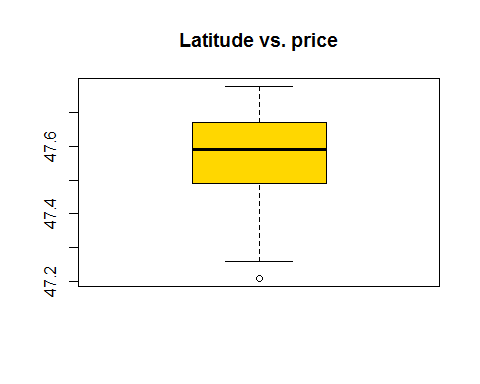
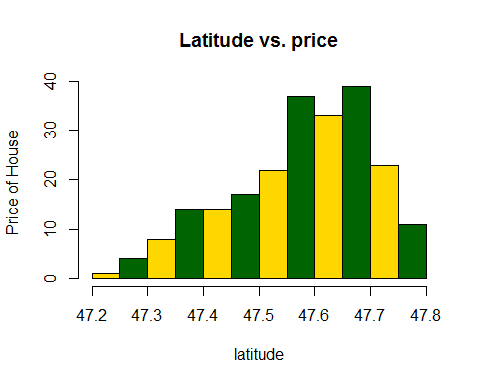
analysis(houseData,18,c('Latitude vs. price','latitude', 'Price of House'), 'N')



## latitudeLog of Price of House

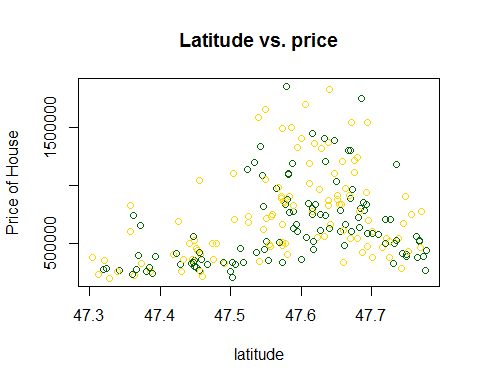


## Latitude vs. price

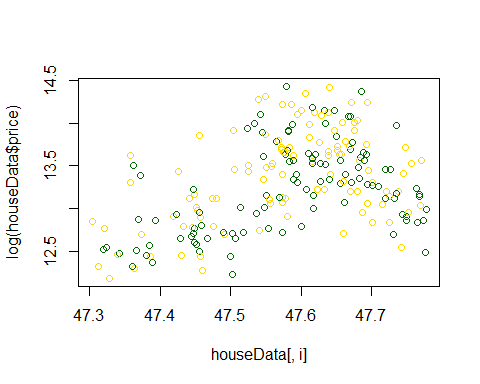


## [1] 0.3502248

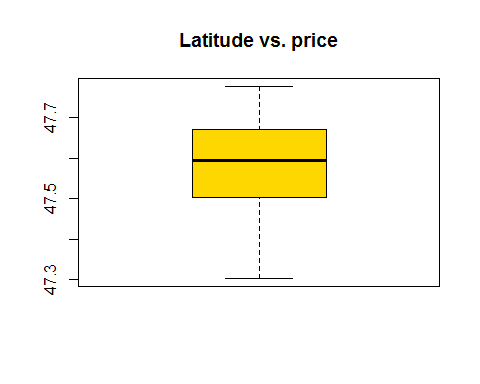
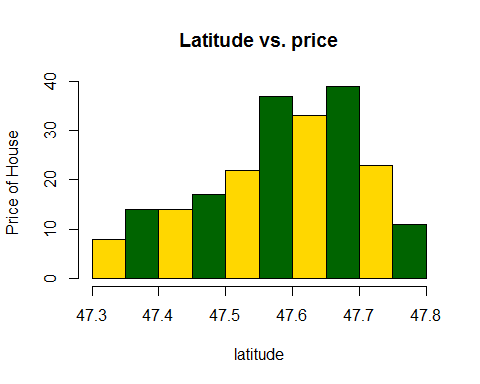
## Price vs. Lat ->> This is more like a normal dist relationship, price peaks around when lat= 47.64 and declines afterwards, but this can be modeled easily. I would say Lat explains the price as well.  
#\*\*\*\*\*\*\*Removing the outliers  
houseData<-subset(houseData,lat>=47.3)  
analysis(houseData,18,c('Latitude vs. price','latitude', 'Price of House'), 'N')



## latitudeLog of Price of House



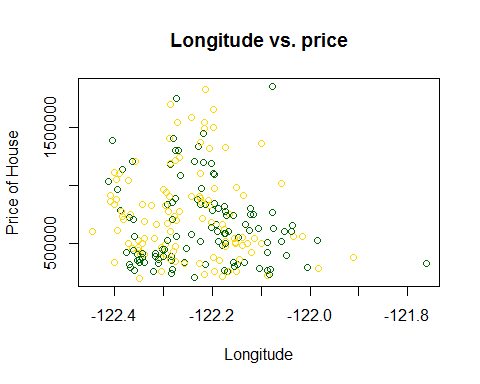
## Latitude vs. price



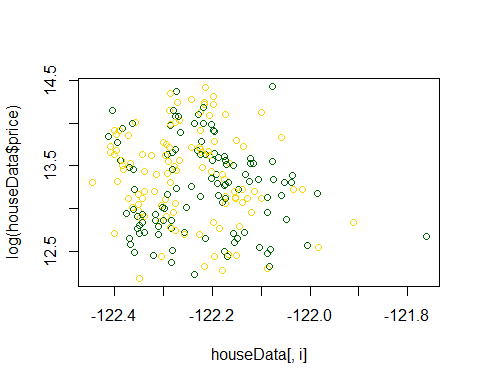
## [1] 0.31377

## LONG Vs Price analysis

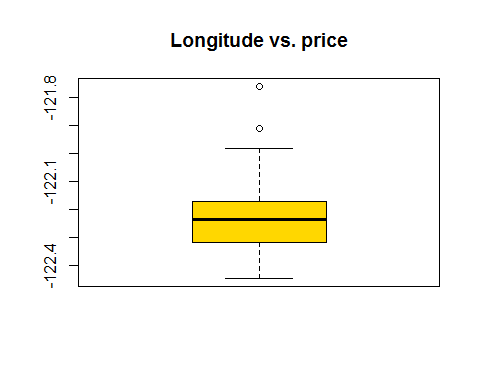
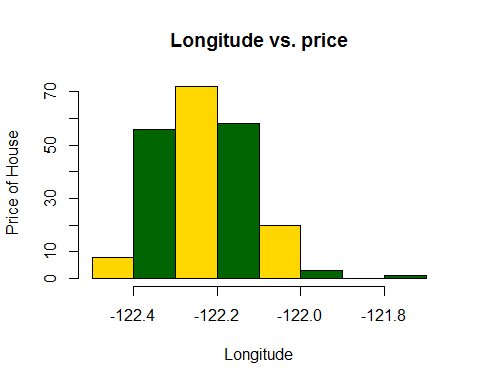
analysis(houseData,19,c('Longitude vs. price','Longitude', 'Price of House'), 'N')



## LongitudeLog of Price of House

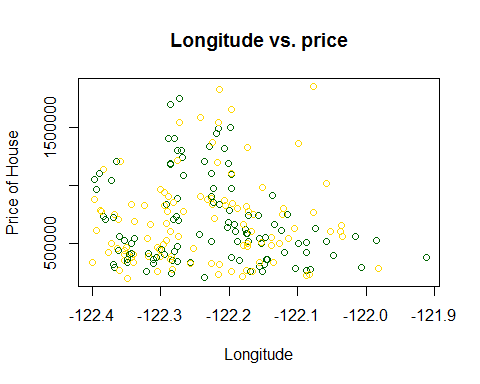


## Longitude vs. price

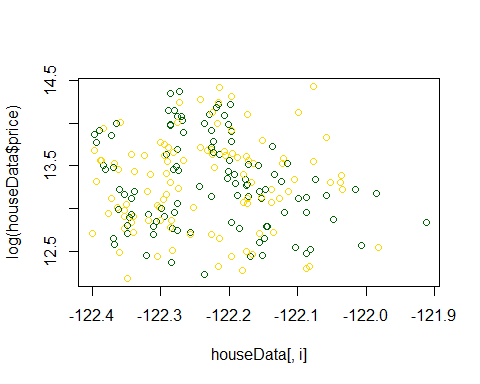


## [1] -0.1101141

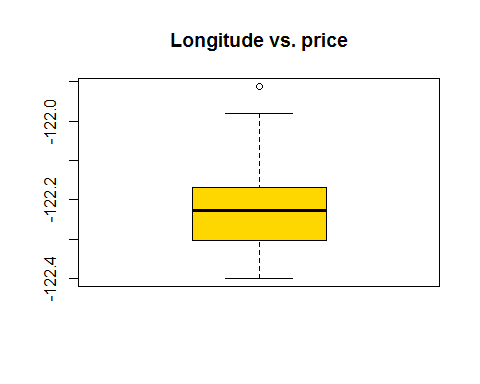
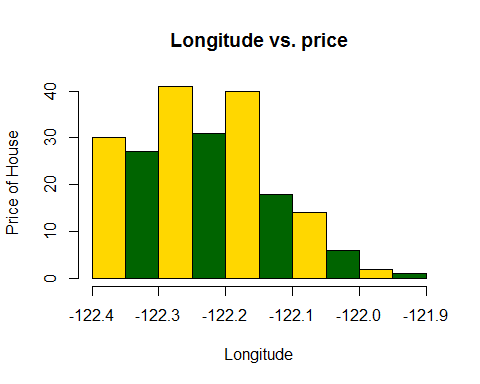
#\*\*\*\*\*\*\*Removing the outliers  
houseData<-subset(houseData,long>=-122.4 & long < -121.8)  
analysis(houseData,19,c('Longitude vs. price','Longitude', 'Price of House'), 'N')



## LongitudeLog of Price of House



## Longitude vs. price



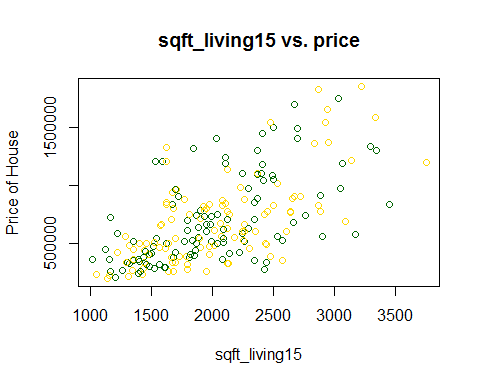
## [1] -0.05926148

## SQFT\_LIVING15 Vs Price analysis

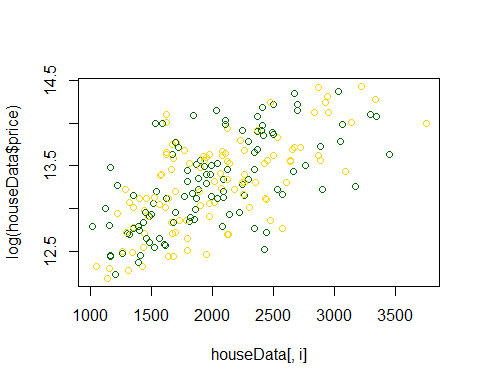
bucketByColumn(houseData,20)

## Min-Max value for: sqft\_living15 , MAX: 1010, MIN: 3750

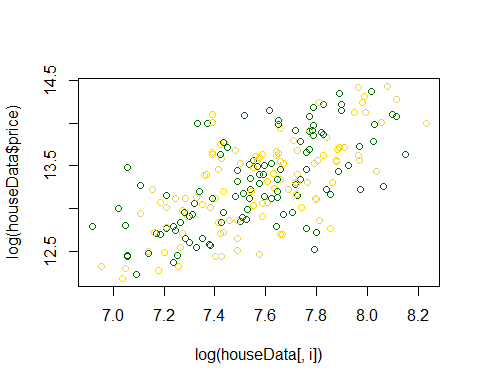
analysis(houseData,20,c('sqft\_living15 vs. price','sqft\_living15', 'Price of House'), 'Y')



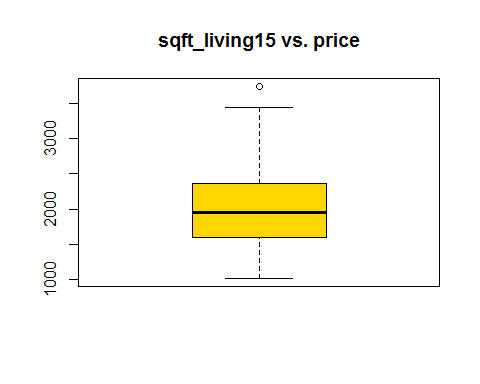
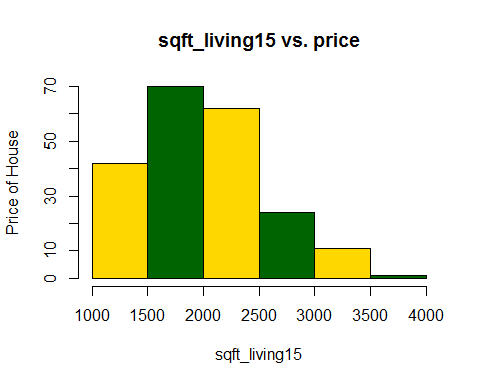
## sqft\_living15Log of Price of House



## sqft\_living15 vs. priceLog of sqft\_living15Log of Price of House



## Log sqft\_living15 vs. price



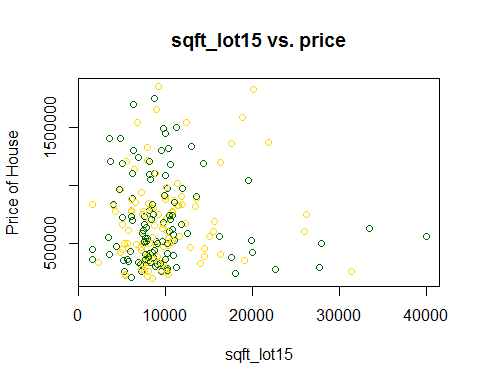
## [1] 0.6262834

## SQFT\_LOT15 Vs Price analysis

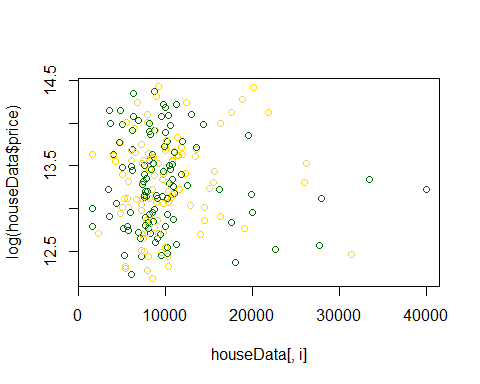
bucketByColumn(houseData,21)

## Min-Max value for: sqft\_lot15 , MAX: 1517, MIN: 39921

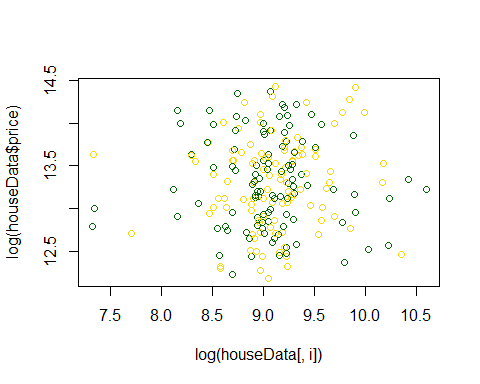
analysis(houseData,21,c('sqft\_lot15 vs. price','sqft\_lot15', 'Price of House'), 'Y')



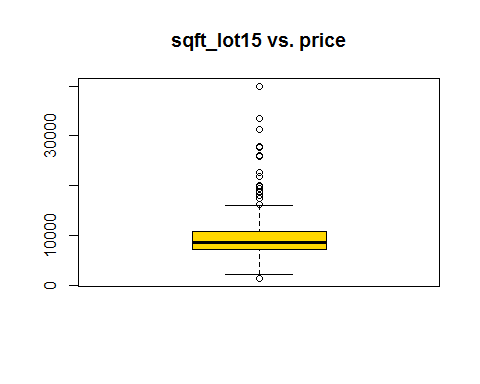
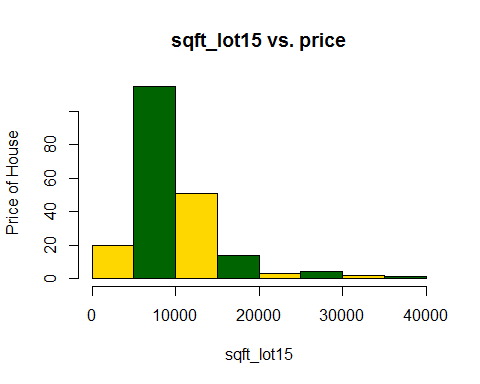
## sqft\_lot15Log of Price of House



## sqft\_lot15 vs. priceLog of sqft\_lot15Log of Price of House

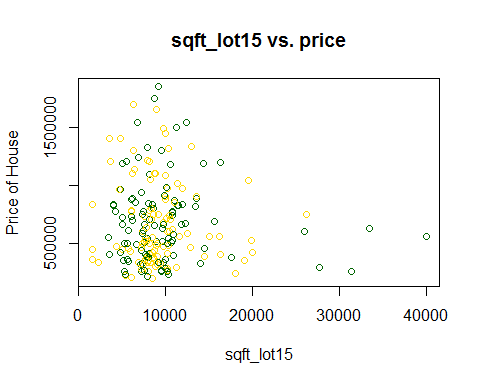


## Log sqft\_lot15 vs. price

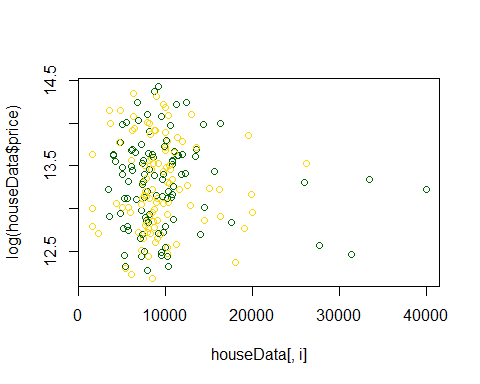


## [1] 0.007371907

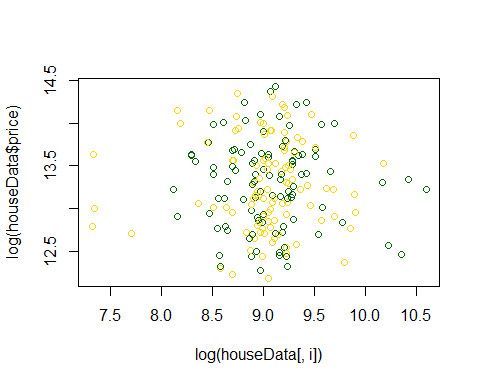
#\*\*\*\*\*\*\*Removing the outliers  
houseData<-subset(houseData,sqft\_lot15>=0 & sqft\_lot<=20000)  
analysis(houseData,21,c('sqft\_lot15 vs. price','sqft\_lot15', 'Price of House'), 'Y')



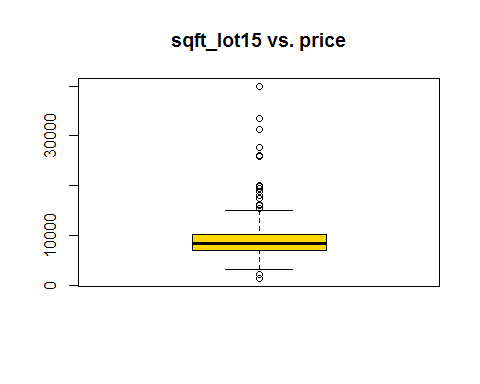
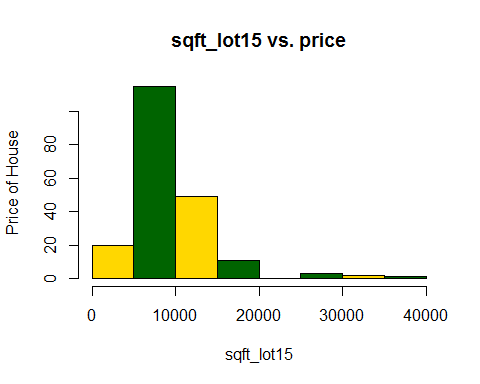
## sqft\_lot15Log of Price of House



## sqft\_lot15 vs. priceLog of sqft\_lot15Log of Price of House



## Log sqft\_lot15 vs. price



## [1] -0.06148357

bucketByColumn(houseData,21)

## Min-Max value for: sqft\_lot15 , MAX: 1517, MIN: 39921

## My fisrt 5 variables are: sqft\_living, bathrooms, grade, view and lat.

## 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]

## Plots 1,2 and 3 shows the correlation between each variables and they are:

# 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 sqft\_lot: 0.089660861

# 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 condition: 0.036361789

# corr between price vs grade: 0.66743426

# corr between price vs sqft\_above: 0.6055672984

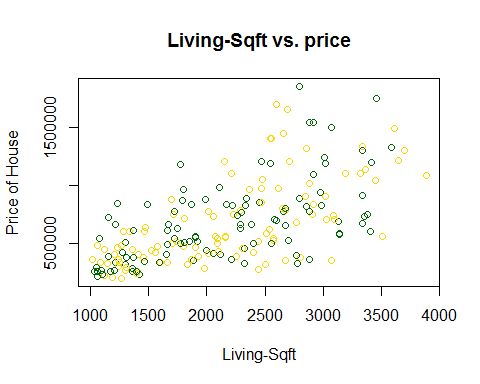
# corr between price vs yr\_built: 0.05401153

# corr between price vs lat: 0.3070034800

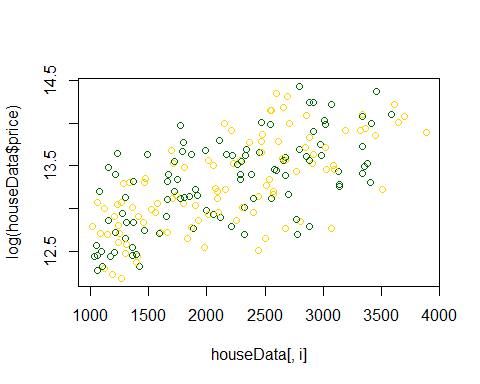
# corr between price vs long: 0.02162624

## Start with price & sqft\_living

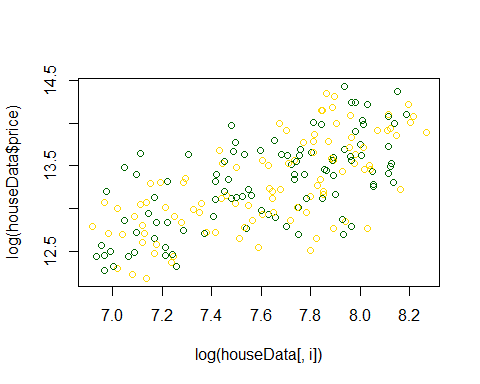
analysis(houseData,6,c('Living-Sqft vs. price','Living-Sqft', 'Price of House'), 'Y')



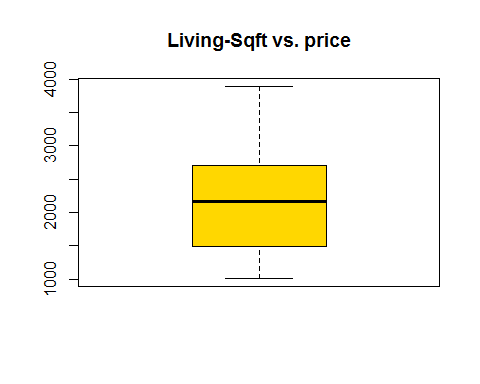
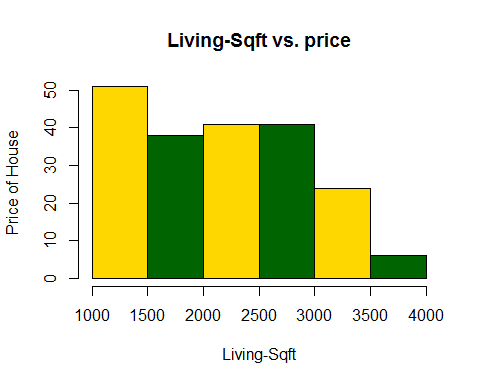
## Living-SqftLog of Price of House



## Living-Sqft vs. priceLog of Living-SqftLog of Price of House

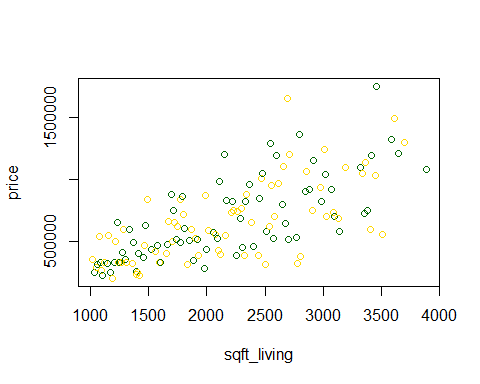


## Log Living-Sqft vs. price

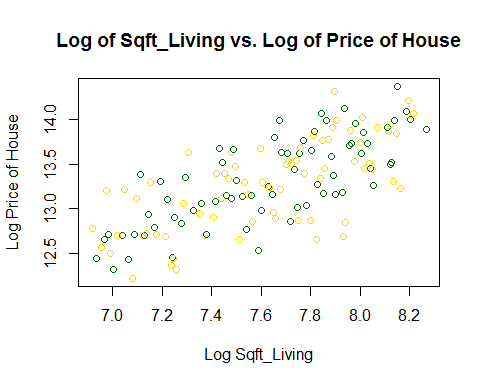


## [1] 0.6308966

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



scatterplot1<-recordPlot()  
  
## Plot does not show that price and sqft\_living are linearly related. It looks like an exponential relationship.  
  
## I am using aggregated data as opposed to using the raw data. By "aggragated" data, I mean I take the mean for all the same sqft\_living.  
  
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()

## 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, condition, grade, yr built, lat, long

## Start with one variable predication  
## Creating Models using 1 variables fpr each of the variables with price, so total we have 12 Models.   
rn\_train <- sample(nrow(houseData),floor(nrow(houseData)\*0.60))  
train <- houseData[rn\_train,colnames(houseData)]  
test <- houseData[-rn\_train,colnames(houseData)]  
  
SSEVals <- list(c(), c())  
findSSEByIndesx <- function(i){  
modlm<-lm(as.formula(paste("log(price)~", paste(c(colnames(houseData)[i]), collapse="+"))),data=train)  
 predt<-exp(predict(modlm,newdata=test))  
 SSE<-sum((test$price-predt)^2)  
 SSEVals[1]<-c(SSEVals[1],i)  
 SSEVals[2]<-c(SSEVals[2],SSE)  
}  
  
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)  
}  
  
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('condition', SSEVals)  
SSEVals = findSSEByColName('grade', SSEVals)  
SSEVals = findSSEByColName('yr\_built', SSEVals)  
SSEVals = findSSEByColName('lat', SSEVals)  
SSEVals = findSSEByColName('long', SSEVals)  
SSEArr = SSEVals[[2]];  
which(SSEArr==min(SSEArr))

## [1] 9

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

## [1] "grade"

## Conculsion: SSE is minimum for variable grade so it is the best predictor, when we use single variable.

# 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  
#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  
  
model01 <- lm(data=train,log(price)~grade)  
r\_squared\_model01<-summary(model01)$r.squared  
  
  
# Now let us add more variable which has greater impact on the price prediction. Let us call it model02. here i have selecetd log(sqft\_living), bedrooms, bathrooms, grade, waterfront  
  
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   
## -0.59831 -0.18336 -0.00168 0.16796 0.61502   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 7.95141 0.71700 11.090 < 2e-16 \*\*\*  
## log(sqft\_living) 0.40219 0.11359 3.541 0.000579 \*\*\*  
## bedrooms -0.05164 0.04015 -1.286 0.200985   
## bathrooms 0.19661 0.05937 3.312 0.001243 \*\*   
## grade 0.25679 0.03154 8.143 5.43e-13 \*\*\*  
## waterfront 0.40502 0.20449 1.981 0.050037 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.2762 on 114 degrees of freedom  
## Multiple R-squared: 0.7173, Adjusted R-squared: 0.7049   
## F-statistic: 57.86 on 5 and 114 DF, p-value: < 2.2e-16

r\_squared\_model02<-summary(model02)$r.squared  
  
cat("\nR-Squared for Model-02 is ",100\*(r\_squared\_model02/r\_squared\_model01-1),"% better than Model-01.\nR-squared for Model-02 and Model-01 are:", r\_squared\_model02, "and", r\_squared\_model01, "respectively.")

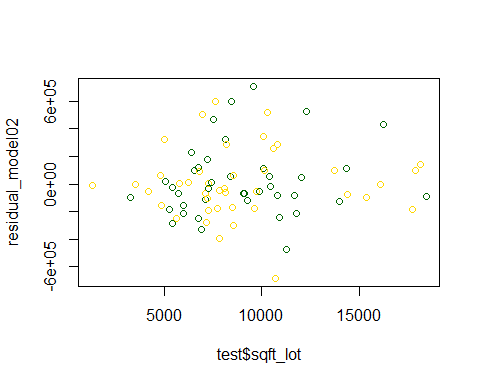
##   
## R-Squared for Model-02 is 25.41836 % better than Model-01.  
## R-squared for Model-02 and Model-01 are: 0.7173326 and 0.5719518 respectively.

##compute RMSE for Model-02  
predic\_model02<-exp(predict(model02,newdata=test))   
RMSE\_model02=sqrt(sum((predic\_model02 - test$price)^2)/nrow(test))

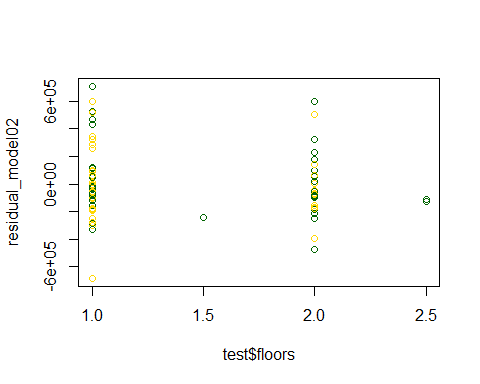
## If we plot residual vs. a variable that is not used in the prediction and if we see any recognizable patterns, thenit indicates that some of the variation in residual is due to non-used variable therefore we should include it in our model to reduce the residual errors.

## To calculate residuals, we simply need to substract predic\_model02 from the actual price.

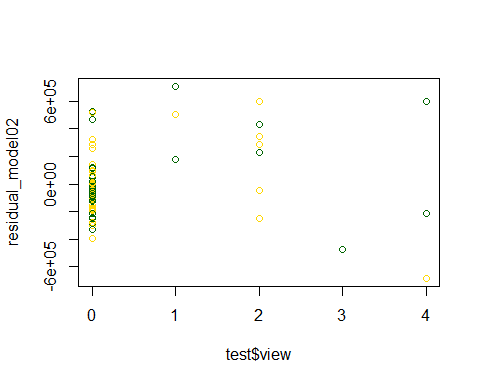
# here we are trying to find more eligible cariable which can impact prediction of price  
  
residual\_model02=test$price - predic\_model02  
## Residual vs. sqft\_lot  
plot(test$sqft\_lot,residual\_model02, col=(c("gold","darkgreen")))



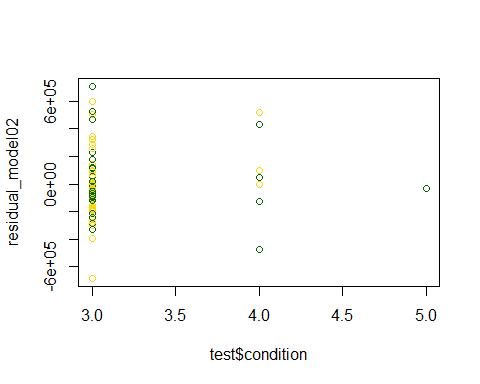
## Residual vs. floors  
plot(test$floors,residual\_model02, col=(c("gold","darkgreen")))



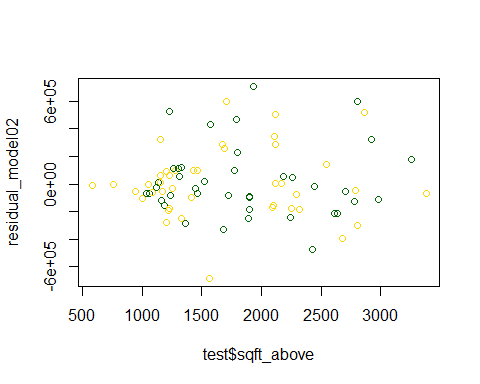
## Residual vs. view  
plot(test$view,residual\_model02, col=(c("gold","darkgreen")))



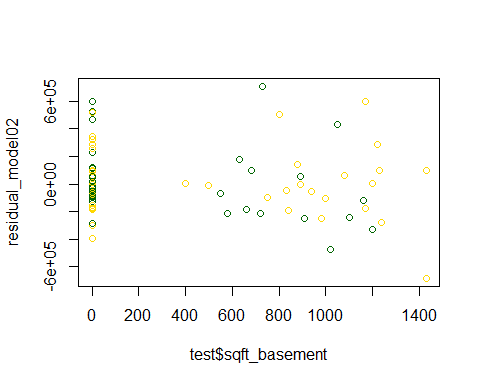
## Residual vs. condition  
plot(test$condition,residual\_model02, col=(c("gold","darkgreen")))



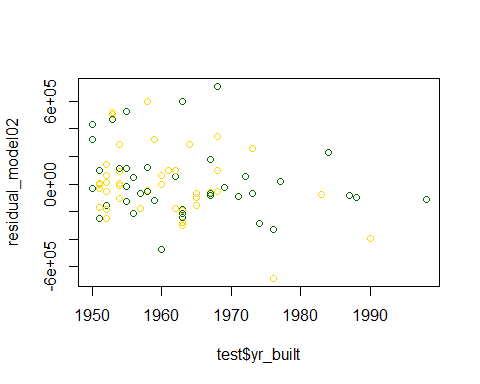
## Residual vs. sqft\_above  
plot(test$sqft\_above,residual\_model02, col=(c("gold","darkgreen")))



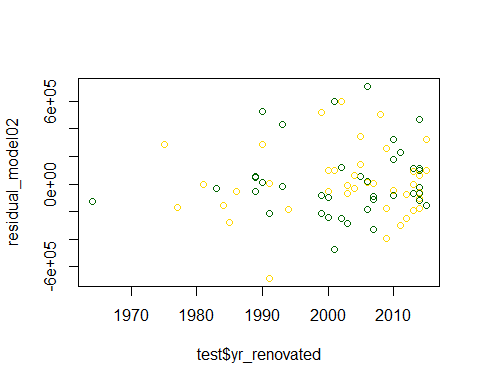
## Residual vs. sqft\_basement  
plot(test$sqft\_basement,residual\_model02, col=(c("gold","darkgreen")))



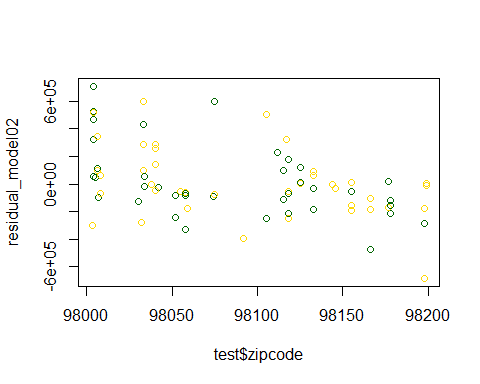
## Residual vs. yr\_built  
plot(test$yr\_built,residual\_model02, col=(c("gold","darkgreen")))



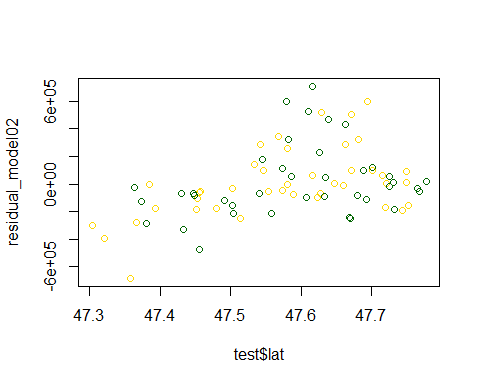
## Residual vs. yr\_renovated  
plot(test$yr\_renovated,residual\_model02, col=(c("gold","darkgreen")))



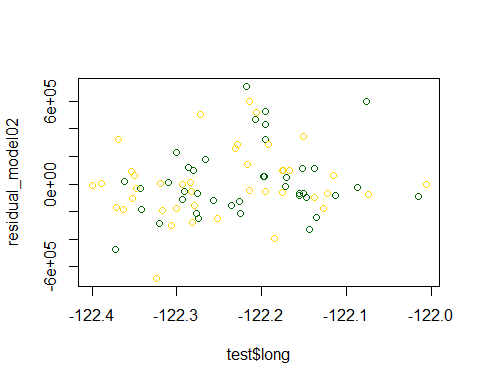
## Residual vs. zipcode  
plot(test$zipcode,residual\_model02, col=(c("gold","darkgreen")))



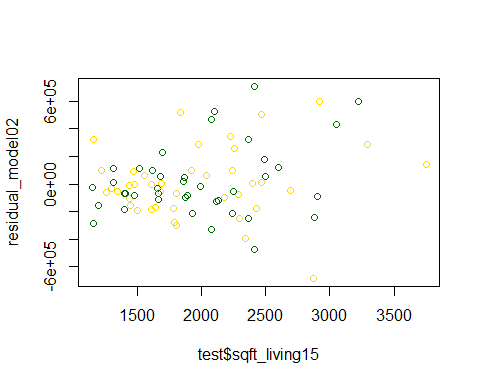
## Residual vs. lat  
plot(test$lat,residual\_model02, col=(c("gold","darkgreen")))



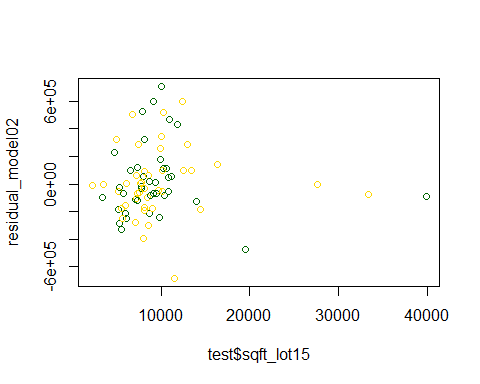
## Residual vs. long  
plot(test$long,residual\_model02, col=(c("gold","darkgreen")))



## 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("\nConclusion: After analzing scattered plot, we found that yr\_built & lat are good candidate to predict the price of house")

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

#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   
## -0.68977 -0.14740 -0.02087 0.17411 0.59416   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -45.275780 11.107332 -4.076 8.59e-05 \*\*\*  
## log(sqft\_living) 0.380325 0.100294 3.792 0.000242 \*\*\*  
## bedrooms -0.026694 0.035298 -0.756 0.451089   
## bathrooms 0.142087 0.053360 2.663 0.008891 \*\*   
## grade 0.249099 0.027955 8.911 1.07e-14 \*\*\*  
## waterfront 0.397750 0.178640 2.227 0.027979 \*   
## yr\_built -0.001627 0.002422 -0.672 0.503118   
## lat 1.191205 0.199414 5.974 2.80e-08 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.2412 on 112 degrees of freedom  
## Multiple R-squared: 0.7883, Adjusted R-squared: 0.7751   
## F-statistic: 59.58 on 7 and 112 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 9.894937 % better than Model-02.  
## R-squared for Model-03 and Model-02 are: 0.7883122 and 0.7173326 respectively.So here Model-03 wins over other previous model.

## RMSE for Model-03:  
predic\_model03<-exp(predict(model03,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: 249039.3   
## RMSE for model03: 218744.4

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 13.85 % more than Model-03. So Model-03 predicts the prices better.

#log(price) seems to have good correlation with exp(bathrooms)  
#log(price) seems to have good correlation with log(lat). to reduce the noice and we will use lat-min(lat), 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  
  
modelComplex<-lm(log(price)~log(sqft\_living)+log(bedrooms+0.5)+exp(bathrooms)+grade+waterfront+log(abs(lat-min(lat))+0.5)+log(abs(long-min(long))+0.05)+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(modelComplex)

##   
## Call:  
## lm(formula = log(price) ~ log(sqft\_living) + log(bedrooms + 0.5) +   
## exp(bathrooms) + grade + waterfront + log(abs(lat - min(lat)) +   
## 0.5) + log(abs(long - min(long)) + 0.05) + 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   
## -0.44586 -0.13026 -0.03403 0.15440 0.49628   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 1.920e+00 5.321e+00 0.361 0.7191  
## log(sqft\_living) 4.833e-01 6.657e-01 0.726 0.4696  
## log(bedrooms + 0.5) 9.472e-01 1.106e+00 0.857 0.3937  
## exp(bathrooms) 4.218e-05 6.991e-03 0.006 0.9952  
## grade -3.131e-01 3.295e-01 -0.950 0.3443  
## waterfront 4.088e-01 2.573e-01 1.589 0.1154  
## log(abs(lat - min(lat)) + 0.5) 8.920e-01 1.438e-01 6.204 1.35e-08  
## log(abs(long - min(long)) + 0.05) 8.100e-02 4.227e-02 1.916 0.0583  
## log(view + 0.5) -2.458e-02 1.696e-01 -0.145 0.8851  
## condition -1.736e-02 2.089e-01 -0.083 0.9339  
## log(sqft\_above + 0.05) 2.349e-01 2.513e-01 0.935 0.3521  
## log(sqft\_basement + 0.05) 7.952e-03 1.302e-02 0.611 0.5427  
## log(sqft\_lot15) -1.007e-01 5.139e-02 -1.959 0.0530  
## log(2015 - yr\_renovated + 1) -5.775e-02 2.532e-02 -2.281 0.0248  
## bedrooms -1.221e-01 1.428e+00 -0.086 0.9320  
## bathrooms -2.456e-01 2.912e-01 -0.843 0.4012  
## log(grade) 4.842e+00 2.689e+00 1.801 0.0749  
## exp(condition) 6.022e-02 2.934e-02 2.053 0.0428  
## view 1.279e-01 1.627e-01 0.786 0.4339  
## bedrooms:bathrooms 9.054e-02 8.506e-02 1.064 0.2898  
## log(grade):exp(condition) -2.824e-02 1.378e-02 -2.049 0.0432  
## log(sqft\_living):bedrooms -4.561e-02 1.801e-01 -0.253 0.8006  
## bedrooms:view -1.742e-02 3.962e-02 -0.440 0.6612  
##   
## (Intercept)   
## log(sqft\_living)   
## log(bedrooms + 0.5)   
## exp(bathrooms)   
## grade   
## waterfront   
## log(abs(lat - min(lat)) + 0.5) \*\*\*  
## log(abs(long - min(long)) + 0.05) .   
## 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   
## 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: 0.2268 on 97 degrees of freedom  
## Multiple R-squared: 0.8379, Adjusted R-squared: 0.8011   
## F-statistic: 22.79 on 22 and 97 DF, p-value: < 2.2e-16

## computing RMSE for Model-Complex  
predic\_modelComplex<-round(exp(predict(modelComplex,newdata=test)),0)   
RMSE\_modelComplex=sqrt(sum((predic\_modelComplex - test$price)^2)/nrow(test))   
  
cat("RMSE for model03:",RMSE\_model03,"\nRMSE for modelComplex:",RMSE\_modelComplex)

## RMSE for model03: 218744.4   
## RMSE for modelComplex: 213337.8

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

## Conclusion: RMSE for Model-03 is 2.53 % more than Model-Complex, so here Model-Complex is the clear winner.

## Selection Method: End

# houseData regression Model

# once above model become winner, let us run the model on different set of test and training data and find the best Model Equestion.

# I took 60% for the training and 40 % for the testing dataset.

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))  
i=0.6  
storage <- list(c(), c(), c(),c())  
for(i in seq(from=0.60, to=0.95, by=0.02)){  
 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)~log(sqft\_living)+log(bedrooms+0.5)+exp(bathrooms)+grade+waterfront+log(abs(lat-min(lat))+0.2)+log(abs(long-min(long))+0.01)+log(view+0.5)+condition+log(sqft\_above+0.05)+log(sqft\_basement+0.001)+log(sqft\_lot15)+log(2015-yr\_renovated+1)+(bedrooms\*bathrooms)+(log(grade)\*exp(condition))+(bedrooms\*log(sqft\_living))+(view\*bedrooms),data=train)  
   
 prediction <- round(exp(predict(model,newdata = test)),0)  
 train\_prediction = fitted(model)  
 training\_rmse = sqrt(sum((train\_prediction-train$price)^2)/nrow(train))  
 testing\_rmse = sqrt(sum((prediction - test$price)^2)/nrow(test))  
 cat("\r\n", i)  
 print(model)  
 storage[[1]]<-c(storage[[1]],prediction)  
 storage[[2]]<-c(storage[[2]],testing\_rmse)  
 storage[[3]]<-c(storage[[3]],test)  
 storage[[4]]<-c(storage[[4]],training\_rmse)  
}

##   
## 0.6  
## Call:  
## lm(formula = log(price) ~ log(sqft\_living) + log(bedrooms + 0.5) +   
## exp(bathrooms) + grade + waterfront + log(abs(lat - min(lat)) +   
## 0.2) + log(abs(long - min(long)) + 0.01) + log(view + 0.5) +   
## condition + log(sqft\_above + 0.05) + log(sqft\_basement +   
## 0.001) + log(sqft\_lot15) + log(2015 - yr\_renovated + 1) +   
## (bedrooms \* bathrooms) + (log(grade) \* exp(condition)) +   
## (bedrooms \* log(sqft\_living)) + (view \* bedrooms), data = train)  
##   
## Coefficients:  
## (Intercept) log(sqft\_living)   
## 4.7630074 0.6371260   
## log(bedrooms + 0.5) exp(bathrooms)   
## 0.1463986 -0.0024088   
## grade waterfront   
## 0.1453495 0.4705665   
## log(abs(lat - min(lat)) + 0.2) log(abs(long - min(long)) + 0.01)   
## 0.6867094 0.0697492   
## log(view + 0.5) condition   
## 0.2211488 -0.0372482   
## log(sqft\_above + 0.05) log(sqft\_basement + 0.001)   
## 0.5275727 0.0189151   
## log(sqft\_lot15) log(2015 - yr\_renovated + 1)   
## -0.0448680 -0.0865976   
## bedrooms bathrooms   
## 0.8985542 -0.6341218   
## log(grade) exp(condition)   
## 0.7163363 0.0220626   
## view bedrooms:bathrooms   
## -0.0599290 0.1907324   
## log(grade):exp(condition) log(sqft\_living):bedrooms   
## -0.0108036 -0.1786226   
## bedrooms:view   
## 0.0007922   
##   
##   
## 0.62  
## Call:  
## lm(formula = log(price) ~ log(sqft\_living) + log(bedrooms + 0.5) +   
## exp(bathrooms) + grade + waterfront + log(abs(lat - min(lat)) +   
## 0.2) + log(abs(long - min(long)) + 0.01) + log(view + 0.5) +   
## condition + log(sqft\_above + 0.05) + log(sqft\_basement +   
## 0.001) + log(sqft\_lot15) + log(2015 - yr\_renovated + 1) +   
## (bedrooms \* bathrooms) + (log(grade) \* exp(condition)) +   
## (bedrooms \* log(sqft\_living)) + (view \* bedrooms), data = train)  
##   
## Coefficients:  
## (Intercept) log(sqft\_living)   
## 4.197438 0.608436   
## log(bedrooms + 0.5) exp(bathrooms)   
## 0.025108 0.004191   
## grade waterfront   
## 0.056859 0.157989   
## log(abs(lat - min(lat)) + 0.2) log(abs(long - min(long)) + 0.01)   
## 0.589967 0.097647   
## log(view + 0.5) condition   
## 0.271815 0.055607   
## log(sqft\_above + 0.05) log(sqft\_basement + 0.001)   
## 0.223309 0.001723   
## log(sqft\_lot15) log(2015 - yr\_renovated + 1)   
## -0.064833 -0.063266   
## bedrooms bathrooms   
## 0.465194 -0.280174   
## log(grade) exp(condition)   
## 2.075273 0.052049   
## view bedrooms:bathrooms   
## 0.020075 0.070435   
## log(grade):exp(condition) log(sqft\_living):bedrooms   
## -0.024521 -0.083865   
## bedrooms:view   
## -0.029324

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

##   
## 0.64  
## Call:  
## lm(formula = log(price) ~ log(sqft\_living) + log(bedrooms + 0.5) +   
## exp(bathrooms) + grade + waterfront + log(abs(lat - min(lat)) +   
## 0.2) + log(abs(long - min(long)) + 0.01) + log(view + 0.5) +   
## condition + log(sqft\_above + 0.05) + log(sqft\_basement +   
## 0.001) + log(sqft\_lot15) + log(2015 - yr\_renovated + 1) +   
## (bedrooms \* bathrooms) + (log(grade) \* exp(condition)) +   
## (bedrooms \* log(sqft\_living)) + (view \* bedrooms), data = train)  
##   
## Coefficients:  
## (Intercept) log(sqft\_living)   
## 0.471130 0.296063   
## log(bedrooms + 0.5) exp(bathrooms)   
## 0.073608 -0.001637   
## grade waterfront   
## -0.193996 0.410725   
## log(abs(lat - min(lat)) + 0.2) log(abs(long - min(long)) + 0.01)   
## 0.637186 0.081040   
## log(view + 0.5) condition   
## 0.247749 1.903515   
## log(sqft\_above + 0.05) log(sqft\_basement + 0.001)   
## 0.270804 0.007229   
## log(sqft\_lot15) log(2015 - yr\_renovated + 1)   
## -0.058560 -0.120101   
## bedrooms bathrooms   
## -0.419649 -0.360855   
## log(grade) exp(condition)   
## 3.843916 NA   
## view bedrooms:bathrooms   
## -0.182331 0.106069   
## log(grade):exp(condition) log(sqft\_living):bedrooms   
## -0.023588 0.012758   
## bedrooms:view   
## 0.028626

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

##   
## 0.66  
## Call:  
## lm(formula = log(price) ~ log(sqft\_living) + log(bedrooms + 0.5) +   
## exp(bathrooms) + grade + waterfront + log(abs(lat - min(lat)) +   
## 0.2) + log(abs(long - min(long)) + 0.01) + log(view + 0.5) +   
## condition + log(sqft\_above + 0.05) + log(sqft\_basement +   
## 0.001) + log(sqft\_lot15) + log(2015 - yr\_renovated + 1) +   
## (bedrooms \* bathrooms) + (log(grade) \* exp(condition)) +   
## (bedrooms \* log(sqft\_living)) + (view \* bedrooms), data = train)  
##   
## Coefficients:  
## (Intercept) log(sqft\_living)   
## 2.722285 0.578001   
## log(bedrooms + 0.5) exp(bathrooms)   
## -0.341182 -0.005255   
## grade waterfront   
## 0.151978 0.484222   
## log(abs(lat - min(lat)) + 0.2) log(abs(long - min(long)) + 0.01)   
## 0.626947 0.062478   
## log(view + 0.5) condition   
## 0.281874 0.998111   
## log(sqft\_above + 0.05) log(sqft\_basement + 0.001)   
## 0.526932 0.014230   
## log(sqft\_lot15) log(2015 - yr\_renovated + 1)   
## -0.039500 -0.058215   
## bedrooms bathrooms   
## 0.867771 -0.601319   
## log(grade) exp(condition)   
## 0.730555 NA   
## view bedrooms:bathrooms   
## -0.091680 0.181004   
## log(grade):exp(condition) log(sqft\_living):bedrooms   
## -0.012525 -0.160856   
## bedrooms:view   
## 0.001605   
##   
##   
## 0.68  
## Call:  
## lm(formula = log(price) ~ log(sqft\_living) + log(bedrooms + 0.5) +   
## exp(bathrooms) + grade + waterfront + log(abs(lat - min(lat)) +   
## 0.2) + log(abs(long - min(long)) + 0.01) + log(view + 0.5) +   
## condition + log(sqft\_above + 0.05) + log(sqft\_basement +   
## 0.001) + log(sqft\_lot15) + log(2015 - yr\_renovated + 1) +   
## (bedrooms \* bathrooms) + (log(grade) \* exp(condition)) +   
## (bedrooms \* log(sqft\_living)) + (view \* bedrooms), data = train)  
##   
## Coefficients:  
## (Intercept) log(sqft\_living)   
## 5.463655 0.427441   
## log(bedrooms + 0.5) exp(bathrooms)   
## 0.764670 0.003496   
## grade waterfront   
## 0.238472 0.382517   
## log(abs(lat - min(lat)) + 0.2) log(abs(long - min(long)) + 0.01)   
## 0.579457 0.035124   
## log(view + 0.5) condition   
## 0.247295 0.069942   
## log(sqft\_above + 0.05) log(sqft\_basement + 0.001)   
## 0.378994 0.008604   
## log(sqft\_lot15) log(2015 - yr\_renovated + 1)   
## -0.002694 -0.068041   
## bedrooms bathrooms   
## 0.033240 -0.347598   
## log(grade) exp(condition)   
## 0.467946 0.060394   
## view bedrooms:bathrooms   
## -0.045167 0.093557   
## log(grade):exp(condition) log(sqft\_living):bedrooms   
## -0.028720 -0.061878   
## bedrooms:view   
## -0.010549

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

##   
## 0.7  
## Call:  
## lm(formula = log(price) ~ log(sqft\_living) + log(bedrooms + 0.5) +   
## exp(bathrooms) + grade + waterfront + log(abs(lat - min(lat)) +   
## 0.2) + log(abs(long - min(long)) + 0.01) + log(view + 0.5) +   
## condition + log(sqft\_above + 0.05) + log(sqft\_basement +   
## 0.001) + log(sqft\_lot15) + log(2015 - yr\_renovated + 1) +   
## (bedrooms \* bathrooms) + (log(grade) \* exp(condition)) +   
## (bedrooms \* log(sqft\_living)) + (view \* bedrooms), data = train)  
##   
## Coefficients:  
## (Intercept) log(sqft\_living)   
## -3.3865901 1.1006660   
## log(bedrooms + 0.5) exp(bathrooms)   
## 0.7354223 0.0005287   
## grade waterfront   
## -0.0736561 0.2987779   
## log(abs(lat - min(lat)) + 0.2) log(abs(long - min(long)) + 0.01)   
## 0.6750588 0.0474141   
## log(view + 0.5) condition   
## 0.1700151 1.4373714   
## log(sqft\_above + 0.05) log(sqft\_basement + 0.001)   
## 0.2914299 0.0055828   
## log(sqft\_lot15) log(2015 - yr\_renovated + 1)   
## 0.0024159 -0.0762643   
## bedrooms bathrooms   
## 1.1080671 -0.7149967   
## log(grade) exp(condition)   
## 2.5587100 NA   
## view bedrooms:bathrooms   
## 0.0687245 0.2037447   
## log(grade):exp(condition) log(sqft\_living):bedrooms   
## -0.0186278 -0.2316744   
## bedrooms:view   
## -0.0264843   
##   
##   
## 0.72  
## Call:  
## lm(formula = log(price) ~ log(sqft\_living) + log(bedrooms + 0.5) +   
## exp(bathrooms) + grade + waterfront + log(abs(lat - min(lat)) +   
## 0.2) + log(abs(long - min(long)) + 0.01) + log(view + 0.5) +   
## condition + log(sqft\_above + 0.05) + log(sqft\_basement +   
## 0.001) + log(sqft\_lot15) + log(2015 - yr\_renovated + 1) +   
## (bedrooms \* bathrooms) + (log(grade) \* exp(condition)) +   
## (bedrooms \* log(sqft\_living)) + (view \* bedrooms), data = train)  
##   
## Coefficients:  
## (Intercept) log(sqft\_living)   
## 3.375267 0.819914   
## log(bedrooms + 0.5) exp(bathrooms)   
## -0.201319 -0.003779   
## grade waterfront   
## 0.185062 0.371302   
## log(abs(lat - min(lat)) + 0.2) log(abs(long - min(long)) + 0.01)   
## 0.679809 0.054705   
## log(view + 0.5) condition   
## 0.176317 0.195342   
## log(sqft\_above + 0.05) log(sqft\_basement + 0.001)   
## 0.413719 0.008858   
## log(sqft\_lot15) log(2015 - yr\_renovated + 1)   
## -0.008747 -0.078227   
## bedrooms bathrooms   
## 1.135000 -0.472548   
## log(grade) exp(condition)   
## 0.450027 0.012918   
## view bedrooms:bathrooms   
## -0.103445 0.152713   
## log(grade):exp(condition) log(sqft\_living):bedrooms   
## -0.008056 -0.192791   
## bedrooms:view   
## 0.023300   
##   
##   
## 0.74  
## Call:  
## lm(formula = log(price) ~ log(sqft\_living) + log(bedrooms + 0.5) +   
## exp(bathrooms) + grade + waterfront + log(abs(lat - min(lat)) +   
## 0.2) + log(abs(long - min(long)) + 0.01) + log(view + 0.5) +   
## condition + log(sqft\_above + 0.05) + log(sqft\_basement +   
## 0.001) + log(sqft\_lot15) + log(2015 - yr\_renovated + 1) +   
## (bedrooms \* bathrooms) + (log(grade) \* exp(condition)) +   
## (bedrooms \* log(sqft\_living)) + (view \* bedrooms), data = train)  
##   
## Coefficients:  
## (Intercept) log(sqft\_living)   
## 0.426425 0.657424   
## log(bedrooms + 0.5) exp(bathrooms)   
## -0.521653 0.002377   
## grade waterfront   
## -0.166521 0.293203   
## log(abs(lat - min(lat)) + 0.2) log(abs(long - min(long)) + 0.01)   
## 0.696242 0.064239   
## log(view + 0.5) condition   
## 0.097134 0.333055   
## log(sqft\_above + 0.05) log(sqft\_basement + 0.001)   
## 0.442053 0.011358   
## log(sqft\_lot15) log(2015 - yr\_renovated + 1)   
## 0.012679 -0.094308   
## bedrooms bathrooms   
## 0.801876 -0.429107   
## log(grade) exp(condition)   
## 3.519815 0.030613   
## view bedrooms:bathrooms   
## -0.061578 0.112070   
## log(grade):exp(condition) log(sqft\_living):bedrooms   
## -0.017708 -0.130716   
## bedrooms:view   
## 0.022182   
##   
##   
## 0.76  
## Call:  
## lm(formula = log(price) ~ log(sqft\_living) + log(bedrooms + 0.5) +   
## exp(bathrooms) + grade + waterfront + log(abs(lat - min(lat)) +   
## 0.2) + log(abs(long - min(long)) + 0.01) + log(view + 0.5) +   
## condition + log(sqft\_above + 0.05) + log(sqft\_basement +   
## 0.001) + log(sqft\_lot15) + log(2015 - yr\_renovated + 1) +   
## (bedrooms \* bathrooms) + (log(grade) \* exp(condition)) +   
## (bedrooms \* log(sqft\_living)) + (view \* bedrooms), data = train)  
##   
## Coefficients:  
## (Intercept) log(sqft\_living)   
## -0.679131 0.816547   
## log(bedrooms + 0.5) exp(bathrooms)   
## 0.008592 -0.001721   
## grade waterfront   
## -0.165157 0.416619   
## log(abs(lat - min(lat)) + 0.2) log(abs(long - min(long)) + 0.01)   
## 0.704921 0.028194   
## log(view + 0.5) condition   
## 0.214391 0.198198   
## log(sqft\_above + 0.05) log(sqft\_basement + 0.001)   
## 0.354746 0.009916   
## log(sqft\_lot15) log(2015 - yr\_renovated + 1)   
## 0.031179 -0.069616   
## bedrooms bathrooms   
## 0.974138 -0.478386   
## log(grade) exp(condition)   
## 3.708049 0.059609   
## view bedrooms:bathrooms   
## -0.160149 0.145342   
## log(grade):exp(condition) log(sqft\_living):bedrooms   
## -0.029629 -0.175341   
## bedrooms:view   
## 0.026572   
##   
##   
## 0.78  
## Call:  
## lm(formula = log(price) ~ log(sqft\_living) + log(bedrooms + 0.5) +   
## exp(bathrooms) + grade + waterfront + log(abs(lat - min(lat)) +   
## 0.2) + log(abs(long - min(long)) + 0.01) + log(view + 0.5) +   
## condition + log(sqft\_above + 0.05) + log(sqft\_basement +   
## 0.001) + log(sqft\_lot15) + log(2015 - yr\_renovated + 1) +   
## (bedrooms \* bathrooms) + (log(grade) \* exp(condition)) +   
## (bedrooms \* log(sqft\_living)) + (view \* bedrooms), data = train)  
##   
## Coefficients:  
## (Intercept) log(sqft\_living)   
## -1.434221 0.929503   
## log(bedrooms + 0.5) exp(bathrooms)   
## 0.275225 0.001945   
## grade waterfront   
## -0.264220 0.399186   
## log(abs(lat - min(lat)) + 0.2) log(abs(long - min(long)) + 0.01)   
## 0.664367 0.039148   
## log(view + 0.5) condition   
## 0.297165 0.137183   
## log(sqft\_above + 0.05) log(sqft\_basement + 0.001)   
## 0.355084 0.010158   
## log(sqft\_lot15) log(2015 - yr\_renovated + 1)   
## -0.018086 -0.071972   
## bedrooms bathrooms   
## 1.256109 -0.413079   
## log(grade) exp(condition)   
## 4.219213 0.052153   
## view bedrooms:bathrooms   
## -0.180772 0.122736   
## log(grade):exp(condition) log(sqft\_living):bedrooms   
## -0.025598 -0.215710   
## bedrooms:view   
## 0.020953   
##   
##   
## 0.8  
## Call:  
## lm(formula = log(price) ~ log(sqft\_living) + log(bedrooms + 0.5) +   
## exp(bathrooms) + grade + waterfront + log(abs(lat - min(lat)) +   
## 0.2) + log(abs(long - min(long)) + 0.01) + log(view + 0.5) +   
## condition + log(sqft\_above + 0.05) + log(sqft\_basement +   
## 0.001) + log(sqft\_lot15) + log(2015 - yr\_renovated + 1) +   
## (bedrooms \* bathrooms) + (log(grade) \* exp(condition)) +   
## (bedrooms \* log(sqft\_living)) + (view \* bedrooms), data = train)  
##   
## Coefficients:  
## (Intercept) log(sqft\_living)   
## -2.2084246 0.9570207   
## log(bedrooms + 0.5) exp(bathrooms)   
## -0.5600769 0.0001062   
## grade waterfront   
## -0.3069373 0.2363502   
## log(abs(lat - min(lat)) + 0.2) log(abs(long - min(long)) + 0.01)   
## 0.6064731 0.0487122   
## log(view + 0.5) condition   
## 0.2176630 0.5212263   
## log(sqft\_above + 0.05) log(sqft\_basement + 0.001)   
## 0.4075629 0.0132812   
## log(sqft\_lot15) log(2015 - yr\_renovated + 1)   
## -0.0549732 -0.0749915   
## bedrooms bathrooms   
## 1.7108539 -0.5052195   
## log(grade) exp(condition)   
## 4.3777106 0.0147517   
## view bedrooms:bathrooms   
## -0.1410784 0.1508433   
## log(grade):exp(condition) log(sqft\_living):bedrooms   
## -0.0115936 -0.2531578   
## bedrooms:view   
## 0.0311164   
##   
##   
## 0.82  
## Call:  
## lm(formula = log(price) ~ log(sqft\_living) + log(bedrooms + 0.5) +   
## exp(bathrooms) + grade + waterfront + log(abs(lat - min(lat)) +   
## 0.2) + log(abs(long - min(long)) + 0.01) + log(view + 0.5) +   
## condition + log(sqft\_above + 0.05) + log(sqft\_basement +   
## 0.001) + log(sqft\_lot15) + log(2015 - yr\_renovated + 1) +   
## (bedrooms \* bathrooms) + (log(grade) \* exp(condition)) +   
## (bedrooms \* log(sqft\_living)) + (view \* bedrooms), data = train)  
##   
## Coefficients:  
## (Intercept) log(sqft\_living)   
## 2.850676 0.816121   
## log(bedrooms + 0.5) exp(bathrooms)   
## -0.550730 -0.001806   
## grade waterfront   
## 0.135324 0.192802   
## log(abs(lat - min(lat)) + 0.2) log(abs(long - min(long)) + 0.01)   
## 0.609439 0.041873   
## log(view + 0.5) condition   
## 0.202317 0.226849   
## log(sqft\_above + 0.05) log(sqft\_basement + 0.001)   
## 0.396045 0.013292   
## log(sqft\_lot15) log(2015 - yr\_renovated + 1)   
## 0.012873 -0.066225   
## bedrooms bathrooms   
## 1.416267 -0.354339   
## log(grade) exp(condition)   
## 0.743446 0.010101   
## view bedrooms:bathrooms   
## -0.002747 0.123232   
## log(grade):exp(condition) log(sqft\_living):bedrooms   
## -0.006745 -0.209211   
## bedrooms:view   
## -0.012890   
##   
##   
## 0.84  
## Call:  
## lm(formula = log(price) ~ log(sqft\_living) + log(bedrooms + 0.5) +   
## exp(bathrooms) + grade + waterfront + log(abs(lat - min(lat)) +   
## 0.2) + log(abs(long - min(long)) + 0.01) + log(view + 0.5) +   
## condition + log(sqft\_above + 0.05) + log(sqft\_basement +   
## 0.001) + log(sqft\_lot15) + log(2015 - yr\_renovated + 1) +   
## (bedrooms \* bathrooms) + (log(grade) \* exp(condition)) +   
## (bedrooms \* log(sqft\_living)) + (view \* bedrooms), data = train)  
##   
## Coefficients:  
## (Intercept) log(sqft\_living)   
## 0.8851564 0.7899993   
## log(bedrooms + 0.5) exp(bathrooms)   
## 0.2608960 -0.0008606   
## grade waterfront   
## -0.1703655 0.2873698   
## log(abs(lat - min(lat)) + 0.2) log(abs(long - min(long)) + 0.01)   
## 0.6450936 0.0593784   
## log(view + 0.5) condition   
## 0.1020159 0.2761226   
## log(sqft\_above + 0.05) log(sqft\_basement + 0.001)   
## 0.2710634 0.0085540   
## log(sqft\_lot15) log(2015 - yr\_renovated + 1)   
## -0.0078155 -0.0650751   
## bedrooms bathrooms   
## 0.7285156 -0.4687054   
## log(grade) exp(condition)   
## 3.3480070 0.0233720   
## view bedrooms:bathrooms   
## -0.0016974 0.1432061   
## log(grade):exp(condition) log(sqft\_living):bedrooms   
## -0.0138987 -0.1504212   
## bedrooms:view   
## 0.0078426   
##   
##   
## 0.86  
## Call:  
## lm(formula = log(price) ~ log(sqft\_living) + log(bedrooms + 0.5) +   
## exp(bathrooms) + grade + waterfront + log(abs(lat - min(lat)) +   
## 0.2) + log(abs(long - min(long)) + 0.01) + log(view + 0.5) +   
## condition + log(sqft\_above + 0.05) + log(sqft\_basement +   
## 0.001) + log(sqft\_lot15) + log(2015 - yr\_renovated + 1) +   
## (bedrooms \* bathrooms) + (log(grade) \* exp(condition)) +   
## (bedrooms \* log(sqft\_living)) + (view \* bedrooms), data = train)  
##   
## Coefficients:  
## (Intercept) log(sqft\_living)   
## 2.049048 0.886521   
## log(bedrooms + 0.5) exp(bathrooms)   
## -0.043276 0.007039   
## grade waterfront   
## 0.032415 0.240068   
## log(abs(lat - min(lat)) + 0.2) log(abs(long - min(long)) + 0.01)   
## 0.640781 0.051231   
## log(view + 0.5) condition   
## 0.085890 0.288476   
## log(sqft\_above + 0.05) log(sqft\_basement + 0.001)   
## 0.241323 0.003640   
## log(sqft\_lot15) log(2015 - yr\_renovated + 1)   
## -0.031670 -0.071771   
## bedrooms bathrooms   
## 0.932085 -0.459716   
## log(grade) exp(condition)   
## 1.858667 0.024654   
## view bedrooms:bathrooms   
## -0.005433 0.113145   
## log(grade):exp(condition) log(sqft\_living):bedrooms   
## -0.014556 -0.159057   
## bedrooms:view   
## 0.013471   
##   
##   
## 0.88  
## Call:  
## lm(formula = log(price) ~ log(sqft\_living) + log(bedrooms + 0.5) +   
## exp(bathrooms) + grade + waterfront + log(abs(lat - min(lat)) +   
## 0.2) + log(abs(long - min(long)) + 0.01) + log(view + 0.5) +   
## condition + log(sqft\_above + 0.05) + log(sqft\_basement +   
## 0.001) + log(sqft\_lot15) + log(2015 - yr\_renovated + 1) +   
## (bedrooms \* bathrooms) + (log(grade) \* exp(condition)) +   
## (bedrooms \* log(sqft\_living)) + (view \* bedrooms), data = train)  
##   
## Coefficients:  
## (Intercept) log(sqft\_living)   
## 1.574387 0.628950   
## log(bedrooms + 0.5) exp(bathrooms)   
## 0.122797 0.004904   
## grade waterfront   
## -0.058071 0.324399   
## log(abs(lat - min(lat)) + 0.2) log(abs(long - min(long)) + 0.01)   
## 0.639719 0.067515   
## log(view + 0.5) condition   
## 0.124831 0.346851   
## log(sqft\_above + 0.05) log(sqft\_basement + 0.001)   
## 0.492651 0.014756   
## log(sqft\_lot15) log(2015 - yr\_renovated + 1)   
## -0.035557 -0.076125   
## bedrooms bathrooms   
## 0.879359 -0.595298   
## log(grade) exp(condition)   
## 2.549994 0.030505   
## view bedrooms:bathrooms   
## 0.007218 0.151460   
## log(grade):exp(condition) log(sqft\_living):bedrooms   
## -0.017868 -0.167158   
## bedrooms:view   
## 0.001473   
##   
##   
## 0.9  
## Call:  
## lm(formula = log(price) ~ log(sqft\_living) + log(bedrooms + 0.5) +   
## exp(bathrooms) + grade + waterfront + log(abs(lat - min(lat)) +   
## 0.2) + log(abs(long - min(long)) + 0.01) + log(view + 0.5) +   
## condition + log(sqft\_above + 0.05) + log(sqft\_basement +   
## 0.001) + log(sqft\_lot15) + log(2015 - yr\_renovated + 1) +   
## (bedrooms \* bathrooms) + (log(grade) \* exp(condition)) +   
## (bedrooms \* log(sqft\_living)) + (view \* bedrooms), data = train)  
##   
## Coefficients:  
## (Intercept) log(sqft\_living)   
## 0.911979 0.736905   
## log(bedrooms + 0.5) exp(bathrooms)   
## 0.463190 0.001538   
## grade waterfront   
## -0.109882 0.275829   
## log(abs(lat - min(lat)) + 0.2) log(abs(long - min(long)) + 0.01)   
## 0.632939 0.071012   
## log(view + 0.5) condition   
## 0.085995 0.292673   
## log(sqft\_above + 0.05) log(sqft\_basement + 0.001)   
## 0.345558 0.009097   
## log(sqft\_lot15) log(2015 - yr\_renovated + 1)   
## -0.028759 -0.062470   
## bedrooms bathrooms   
## 0.754715 -0.484329   
## log(grade) exp(condition)   
## 3.024770 0.028788   
## view bedrooms:bathrooms   
## 0.017515 0.133928   
## log(grade):exp(condition) log(sqft\_living):bedrooms   
## -0.016721 -0.155721   
## bedrooms:view   
## 0.005578   
##   
##   
## 0.92  
## Call:  
## lm(formula = log(price) ~ log(sqft\_living) + log(bedrooms + 0.5) +   
## exp(bathrooms) + grade + waterfront + log(abs(lat - min(lat)) +   
## 0.2) + log(abs(long - min(long)) + 0.01) + log(view + 0.5) +   
## condition + log(sqft\_above + 0.05) + log(sqft\_basement +   
## 0.001) + log(sqft\_lot15) + log(2015 - yr\_renovated + 1) +   
## (bedrooms \* bathrooms) + (log(grade) \* exp(condition)) +   
## (bedrooms \* log(sqft\_living)) + (view \* bedrooms), data = train)  
##   
## Coefficients:  
## (Intercept) log(sqft\_living)   
## 0.4131038 0.7543550   
## log(bedrooms + 0.5) exp(bathrooms)   
## -0.2286118 0.0008727   
## grade waterfront   
## -0.0897444 0.3506560   
## log(abs(lat - min(lat)) + 0.2) log(abs(long - min(long)) + 0.01)   
## 0.6889457 0.0534054   
## log(view + 0.5) condition   
## 0.1023217 0.3145701   
## log(sqft\_above + 0.05) log(sqft\_basement + 0.001)   
## 0.4272335 0.0122447   
## log(sqft\_lot15) log(2015 - yr\_renovated + 1)   
## -0.0024920 -0.0712360   
## bedrooms bathrooms   
## 1.1568219 -0.4704293   
## log(grade) exp(condition)   
## 2.8965733 0.0297096   
## view bedrooms:bathrooms   
## -0.0336678 0.1312650   
## log(grade):exp(condition) log(sqft\_living):bedrooms   
## -0.0173741 -0.1872106   
## bedrooms:view   
## 0.0157251   
##   
##   
## 0.94  
## Call:  
## lm(formula = log(price) ~ log(sqft\_living) + log(bedrooms + 0.5) +   
## exp(bathrooms) + grade + waterfront + log(abs(lat - min(lat)) +   
## 0.2) + log(abs(long - min(long)) + 0.01) + log(view + 0.5) +   
## condition + log(sqft\_above + 0.05) + log(sqft\_basement +   
## 0.001) + log(sqft\_lot15) + log(2015 - yr\_renovated + 1) +   
## (bedrooms \* bathrooms) + (log(grade) \* exp(condition)) +   
## (bedrooms \* log(sqft\_living)) + (view \* bedrooms), data = train)  
##   
## Coefficients:  
## (Intercept) log(sqft\_living)   
## 2.963477 0.561481   
## log(bedrooms + 0.5) exp(bathrooms)   
## 0.173036 0.001578   
## grade waterfront   
## -0.073870 0.310707   
## log(abs(lat - min(lat)) + 0.2) log(abs(long - min(long)) + 0.01)   
## 0.620273 0.058732   
## log(view + 0.5) condition   
## 0.142627 0.304377   
## log(sqft\_above + 0.05) log(sqft\_basement + 0.001)   
## 0.321578 0.008367   
## log(sqft\_lot15) log(2015 - yr\_renovated + 1)   
## -0.023540 -0.067447   
## bedrooms bathrooms   
## 0.407020 -0.410842   
## log(grade) exp(condition)   
## 2.620621 0.020196   
## view bedrooms:bathrooms   
## -0.045014 0.117669   
## log(grade):exp(condition) log(sqft\_living):bedrooms   
## -0.012370 -0.099170   
## bedrooms:view   
## 0.011252

##find the LM with minimun training error  
indx = which(storage[[2]]==min(storage[[2]]))  
indx

## [1] 18

cat("\nRMSE of training model after regression:",storage[[4]][indx],"\nRMSE of testing model after regression:",storage[[2]][indx])

##   
## RMSE of training model after regression: 747928.7   
## RMSE of testing model after regression: 117796.9

finalPredict = storage[[1]][indx]  
testData = storage[[3]][indx]

#Now write the Real & Predicted price file for comparision  
output <- (cbind("ID"=testData$id,"Orginal Price"=testData$price,"Predicted Price"=RMSE\_modelComplex))  
write.csv(output, file = "RealPriceVsPredictedAsStaticticalAnalysis.csv", row.names=FALSE)