House Case Study Report

Yimei Chen, Chris Dong, Qian Li, Jing Song

October 6, 2017

Loading the data and any packages

```
options("max.print"=3)
suppressMessages(library(tidyverse))
suppressMessages(library(magrittr))
suppressMessages(library(leaps))
suppressMessages(library(VIM))
suppressMessages(library(car))
suppressMessages(library(Hmisc))
suppressMessages(library(glmnet))
suppressMessages(library(grid))
suppressMessages(library(gridExtra))
suppressMessages(library(ggcorrplot))
suppressMessages(library(olsrr))
house <- read_csv("housing.txt", col_types = cols())
names(house) <- tolower(names(house))
house0 <- house
```

Convert `mssubclass` to factor and check for `NA`s

```
house$mssubclass <- factor(house$mssubclass)
house %>% sapply(function(x) sum(is.na(x))) %>% sort(decreasing = T)
```

```
##      poolqc miscfeature       alley
##        1453        1406        1369
##  [ reached getOption("max.print") -- omitted 78 entries ]
```

Convert numeric variables that have `NA` to 0. Change `garageyrblt` to indicate whether or not the garage was built AFTER the house was built.

```
house$masvnrarea[which(is.na(house$masvnrarea))] <- 0
house$bsmtfintype1[which(is.na(house$bsmtfintype1))] <- 0
house$bsmtfintype2[which(is.na(house$bsmtfintype2))] <- 0
house$garageyrblt <- (house$garageyrblt > house$yearbuilt) * 1
house$garageyrblt[is.na(house$garageyrblt)] <- 0
```

Impute the NA in `lotfrontage`, `electrical` with K-Nearest Neighbors

```
k = round(sqrt(1460*.8) / 2)

house$lotfrontage <- kNN(house, variable = "lotfrontage",  k = k)$lotfrontage
house$electrical <- kNN(house, variable = "electrical",  k = k)$electrical
```

Convert all other `NA`s to "None"

```
house[is.na(house)] <- "None"
```

Make a new variable, `remodel` that indicates whether or not remodeling took place. Remove the `yearremodadd` variable because it is no longer needed. Make a new variable `soldminusbuilt` that indicates the number of years that it took for the house to get sold after getting built.

```r
house$remodel <- T
house[house$yearbuilt == house$yearremodadd,]$remodel <- F
house$remodel <- as.numeric(house$remodel)
house %<>% select(-yearremodadd)

house$soldminusbuilt <- (house$yrsold - house$yearbuilt)
house %<>% select(-yrsold,-yearbuilt)
```

Combine all of the porch variables into one. Remove `id` because it is obviously not important.

```r
house$porcharea <- with(house, openporchsf + enclosedporch +
    `3ssnporch` + screenporch)
house %<>% select(-id)
```

Change `lotshape` to a boolean whether or not it is Regular.

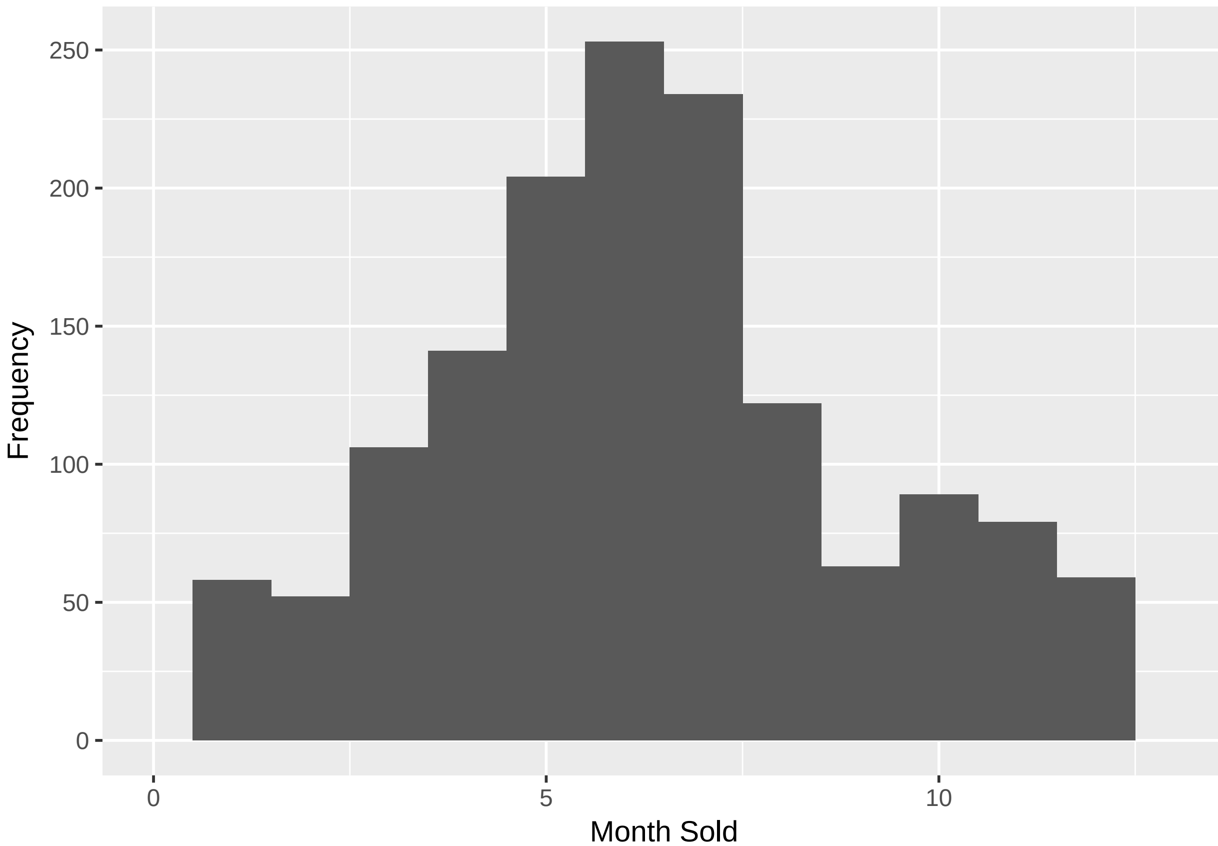
```r
table(house$lotshape)
```

```
##
## IR1 IR2 IR3 Reg
## 484  41  10 925
```

```r
house$lotshape <- (house$lotshape == 'Reg') *1
```

Looking at the histogram of `mosold` we see many more houses being sold near summer time (and part of spring too) so we create a boolean. Most of the time, when we are creating a boolean, it is because it is insignificant otherwise.

```r
house %>% ggplot(aes(x=mosold)) + geom_histogram(binwidth = 1) + xlim(0,13)+
   xlab("Month Sold") +
  ylab("Frequency")
```




```
house$summertime <- (house$mosold %in% 5:7) * 1
```

The next part of the code was very time-consuming but here's the general outline: It is similar to backwards selection but by hand and possibly more thorough because of the refactoring involved rather than simply removing it.

1. Check the p-value and signifiance for a particlar variable.
2. If the variable is numeric and significant, keep it. If the variable is categorical and all levels are significant, keep it. If only some levels are significant then try to bin the factors into smaller number of levels to try and make them statistically significant. If nothing can be done, then remove the variable.
3. Repeat the above steps for the rest of the variables. Each time we remove a variable, we re-run the lm model to check if the Adjusted R Squared changed significantly or not.
4. When we finish going through all the variables, there will be about 30 ones left to consider.

```
house %<>% select(-mosold, -landcontour, -alley, -lotshape)
```

```
house$lotconfig <- (house$lotconfig == "Inside")  * 1
house %<>% select(-lotconfig)
```

Here, we noticed `lotfrontage` became significant when we take the square root. We remove `1stflrsf`, `2ndflrsf`, `lowqualfinsf` because they make up the variable `grlivarea`. At first, we tried having all three of them and deleting `grlivarea` however we found that having just `grlivarea` performed better. We are deleting the porch variables because we have already aggregated them into `porcharea`.

```
fullmodel <- lm(saleprice~sqrt(lotfrontage)+porcharea+.,data = house)
summary(fullmodel)$r.squared
```

```
## [1] 0.9328122
```

```r
house$condition1 <- relevel(factor(house$condition1), ref = "Norm")
house$condition2 <- relevel(factor(house$condition2), ref = "Norm")

house %<>% select(-roofstyle)
house %<>% select(-exterior2nd)

table(house$bldgtype)
```

```
##
##   1Fam 2fmCon Duplex
##   1220     31     52
##  [ reached getOption("max.print") -- omitted 2 entries ]
```

```r
house <- house %>% select(-`1stflrsf`, -`2ndflrsf`, -lowqualfinsf,
    -totalbsmtsf, -openporchsf, -enclosedporch, - `3ssnporch`,
    - screenporch, -garagearea)

house %>% group_by(salecondition) %>% summarise(avgprc = median(saleprice)) %>% arrange(desc(avgprc))
```

```
## # A tibble: 6 x 2
##   salecondition avgprc
##           <chr>  <dbl>
## 1       Partial 244600
## 2        Normal 160000
## 3        Alloca 148145
## 4        Family 140500
## 5       Abnorml 130000
## 6       AdjLand 104000
```

```r
house$salecondition <- (house$salecondition == "Normal") * 1

house %>% group_by(saletype) %>% summarise(avgprc = median(saleprice)) %>% arrange(desc(avgprc))
```

```
## # A tibble: 9 x 2
##   saletype avgprc
##      <chr>  <dbl>
## 1      Con 269600
## 2      New 247453
## 3      CWD 188750
## 4       WD 158000
## 5    ConLw 144000
## 6    ConLD 140000
## 7      COD 139000
## 8    ConLI 125000
## 9      Oth 116050
```

```r
house$newtype <- (house$saletype == 'New') * 1
house <- house %>% select(-saletype)

house$miscfeature <- (house$miscfeature != 'None') * 1
house %<>% select(-miscval, -miscfeature)

house$paveddrive <- (house$paveddrive == 'Y') * 1
house %<>% select(-paveddrive)
```

```r
house$poolqc <- (house$poolqc !="None")*1
house$fence <- (house$fence !="None")*1
```

Here, I am changing the ordered factor into numeric. I want to make a correlation plot with every significant variable so I am converting all variables (as long as it makes sense) to numeric.

```r
house$garagecond <-  as.numeric(factor(house$garagecond,
    levels = c("None","Po","Fa","TA","Gd","Ex"), labels = 0:5))
house$garagequal <-  as.numeric(factor(house$garagequal,
    levels = c("None","Po","Fa","TA","Gd","Ex"), labels = 0:5))

house %<>% select(-fence,-poolqc,-garagecond)

house %>% group_by(garagefinish) %>%
summarise(avgprc = median(saleprice)) %>% arrange(desc(avgprc)) %>% head(2)
```

```
## # A tibble: 2 x 2
##   garagefinish avgprc
##          <chr>  <dbl>
## 1          Fin 215000
## 2          RFn 190000
```

```r
house$garagefinish <-(house$garagefinish == "Fin") *1
house %<>% select(-garagefinish)
```

Here, `fireplacequ` and `fireplaces` are obviously correlated so I choose the one that seems to explain `saleprice` better. However, they both end up being insignificant.

```r
house$fireplacequ <-  as.numeric(factor(house$fireplacequ,
    levels = c("None","Po","Fa","TA","Gd","Ex"), labels = 0:5))
cor(house$saleprice,house$fireplacequ); cor(house$saleprice,house$fireplaces)
```

```
## [1] 0.5204376

## [1] 0.4669288
```

```r
house %<>% select(-fireplacequ, -fireplaces)
```

```r
house %<>% select(-garageyrblt)
house$garagetype <- relevel(factor(house$garagetype), ref = "None")

house$functional <- (house$functional == "Typ") * 1

house$kitchenqual <-  as.numeric(factor(house$kitchenqual,
    levels = c("Po","Fa","TA","Gd","Ex"), labels = 1:5))
```

Similarly, `totrmsabvgrd` is highly correlated with `grlivarea` so I keep the better of the two.

```r
cor(house$totrmsabvgrd ,house$saleprice);cor(house$grlivarea ,house$saleprice)
```

```
## [1] 0.5337232

## [1] 0.7086245
```

```r
house %<>% select(-totrmsabvgrd)
```

I try to combine all of the bath variables but they end up not being significant so I just remove them.

```r
table(house$fullbath)
```

```
##
##   0   1   2   3
##   9 650 768  33
```

```r
house$bath <- house$fullbath + house$halfbath + house$bsmtfullbath + house$bsmthalfbath
house %<>% select(-fullbath,-halfbath, -bsmthalfbath, -bsmtfullbath)
house %<>% select(-bath)
```

```r
house %>% group_by(electrical) %>% summarise(avgprc = median(saleprice)) %>% arrange(desc(avgprc))
```

```
## # A tibble: 5 x 2
##   electrical avgprc
##        <chr>  <dbl>
## 1      SBrkr 170000
## 2      FuseA 121250
## 3      FuseF 115000
## 4      FuseP  82000
## 5        Mix  67000
```

```r
house$electrical <- (house$electrical == "SBrkr") * 1
house %<>% select(-electrical, -centralair)

house$heatingqc <- as.numeric(factor(house$heatingqc,
  levels = c("Po","Fa","TA","Gd","Ex"), labels = 1:5))
table(house$heatingqc)
```

```
##
##   1   2   3
##   1  49 428
##  [ reached getOption("max.print") -- omitted 2 entries ]
```

```r
house$heatingqc <- (house$heatingqc == 5) * 1

house %<>% select(-heating)

table(house$bsmtfintype1)
```

```
##
##   0 ALQ BLQ
##  37 220 148
##  [ reached getOption("max.print") -- omitted 4 entries ]
```

```r
house$bsmtfintype1 <- as.numeric(factor(house$bsmtfintype1,
      levels = c("0","Unf","LwQ","Rec","BLQ","ALQ","GLQ"),
      labels = 0:6))
house$bsmtfintype2 <- as.numeric(factor(house$bsmtfintype2,
      levels = c("0","Unf","LwQ","Rec","BLQ","ALQ","GLQ"),
      labels = 0:6))
house$bsmtfintype1 <- house$bsmtfintype1 + house$bsmtfintype2
house %<>% select(-bsmtfintype1, -bsmtfintype2)


house$bsmtexposure <- relevel(factor(house$bsmtexposure), ref = "None")

table(house$bsmtexposure)
```

```
##
```

```
## None   Av   Gd
##   38  221  134
##  [ reached getOption("max.print") -- omitted 2 entries ]
```

```
house %>% group_by(bsmtexposure) %>% summarise(avgprc = median(saleprice)) %>% arrange(desc(avgprc))
```

```
## # A tibble: 5 x 2
##   bsmtexposure avgprc
##         <fctr>  <dbl>
## 1           Gd 226975
## 2           Av 185850
## 3           Mn 182450
## 4           No 154000
## 5         None 104025
```

```
house$bsmtexposure <- (house$bsmtexposure == "Gd") * 1

house %>% group_by(bsmtcond) %>% summarise(avgprc = median(saleprice)) %>% arrange(desc(avgprc))
```

```
## # A tibble: 5 x 2
##   bsmtcond avgprc
##      <chr>  <dbl>
## 1       Gd 193879
## 2       TA 165000
## 3       Fa 118500
## 4     None 101800
## 5       Po  64000
```

```
table(house$bsmtcond)
```

```
##
##   Fa   Gd None
##   45   65   37
##  [ reached getOption("max.print") -- omitted 2 entries ]
```

```
house$bsmtcond <- as.numeric(factor(house$bsmtcond,
      levels = c("None","Po","Fa","TA","Gd","Ex"),
      labels = 0:5))

house$bsmtqual <- as.numeric(factor(house$bsmtqual,
      levels = c("None","Po","Fa","TA","Gd","Ex"),
      labels = 0:5))
cor(house$bsmtcond,house$bsmtqual)
```

```
## [1] 0.6337134
```

```
cor(house$bsmtcond,house$saleprice);cor(house$bsmtqual,house$saleprice)
```

```
## [1] 0.2126072
```

```
## [1] 0.5852072
```

```
house %<>% select(-bsmtcond)
house %<>% select(-bsmtqual)

table(house$foundation)
```

```
##
## BrkTil CBlock  PConc
```

```
##    146    634    647
##  [ reached getOption("max.print") -- omitted 3 entries ]
```

```
house %>% group_by(foundation) %>% summarise(avgprc = median(saleprice)) %>% arrange(desc(avgprc))
```

```
## # A tibble: 6 x 2
##   foundation avgprc
##        <chr>  <dbl>
## 1      PConc 205000
## 2       Wood 164000
## 3     CBlock 141500
## 4      Stone 126500
## 5     BrkTil 125250
## 6       Slab 104150
```

```
house$foundation <- (house$foundation == "PConc")*1

house$extercond <- as.numeric(factor(house$extercond,
      levels = c("Po","Fa","TA","Gd","Ex"),
      labels = 1:5))
house$exterqual <- as.numeric(factor(house$exterqual,
      levels = c("Po","Fa","TA","Gd","Ex"),
      labels = 1:5))
cor(house$extercond,house$exterqual)
```

```
## [1] 0.00918398
```

```
house$masvnrtype <- relevel(factor(house$masvnrtype), ref = "None")

table(house$masvnrtype)
```

```
##
##    None  BrkCmn BrkFace   Stone
##     872      15     445     128
```

```
house$masvnrtype <- (house$masvnrtype != "None") * 1
```

Boolean whether or not housestyle is either `2Story` or `2.5Fin`.

```
table(house$housestyle)
```

```
##
## 1.5Fin 1.5Unf 1Story
##    154     14    726
##  [ reached getOption("max.print") -- omitted 5 entries ]
```

```
house %>%  group_by(housestyle) %>% summarise(avgprc = median(saleprice)) %>% arrange(desc(avgprc))
```

```
## # A tibble: 8 x 2
##   housestyle avgprc
##        <chr>  <dbl>
## 1     2.5Fin 194000
## 2     2Story 190000
## 3       SLvl 164500
## 4     1Story 154750
## 5     SFoyer 135960
## 6     2.5Unf 133900
## 7     1.5Fin 132000
```

```
## 8     1.5Unf 111250
```

```r
house$housestyle <- (house$housestyle == "2Story" |
                     house$housestyle == "2.5Fin")*1

table(house$bldgtype)
```

```
##
##   1Fam 2fmCon Duplex
##   1220     31     52
##  [ reached getOption("max.print") -- omitted 2 entries ]
```

```r
house %>%  group_by(bldgtype) %>% summarise(avgprc = median(saleprice)) %>% arrange(desc(avgprc))
```

```
## # A tibble: 5 x 2
##   bldgtype avgprc
##      <chr>  <dbl>
## 1   TwnhsE 172200
## 2     1Fam 167900
## 3    Twnhs 137500
## 4   Duplex 135980
## 5   2fmCon 127500
```

```r
house$bldgtype <- (house$bldgtype == "1Fam" | house$bldgtype == "2FmCon") * 1
house %<>% select(-bldgtype)

table(house$landslope)
```

```
##
##  Gtl  Mod  Sev
## 1382   65   13
```

```r
house$landslope <- (house$landslope == "Gtl") * 1
house %<>% select(-landslope)

table(house$utilities)
```

```
##
## AllPub NoSeWa
##   1459      1
```

```r
house %<>% select(-utilities, -street)

house %>%  group_by(mszoning) %>% summarise(avgprc = median(saleprice)) %>% arrange(desc(avgprc))
```

```
## # A tibble: 5 x 2
##   mszoning avgprc
##      <chr>  <dbl>
## 1       FV 205950
## 2       RL 174000
## 3       RH 136500
## 4       RM 120500
## 5  C (all)  74700
```

```r
table(house$mszoning)
```

```
##
## C (all)      FV      RH
```

```
##      10      65      16
##  [ reached getOption("max.print") -- omitted 2 entries ]
```

```
house$mszoning <- relevel(factor(house$mszoning), ref = "RL")

house %<>% select(-mszoning)

house %>%  group_by(mssubclass) %>% summarise(avgprc = median(saleprice)) %>% arrange(desc(avgprc))
```

```
## # A tibble: 15 x 2
##    mssubclass avgprc
##        <fctr>  <dbl>
##  1         60 215200
##  2        120 192000
##  3         80 166500
##  4         75 163500
##  5         20 159250
##  6         70 156000
##  7        160 146000
##  8         40 142500
##  9         85 140750
## 10         90 135980
## 11         50 132000
## 12        190 128250
## 13         45 107500
## 14         30  99900
## 15        180  88500
```

```
house %<>% select(-mssubclass, -lotfrontage, -porcharea, -extercond,-foundation,
                  -exterior1st)


house %>%  group_by(condition1) %>% summarise(avgprc = median(saleprice)) %>% arrange(desc(avgprc))
```

```
## # A tibble: 9 x 2
##   condition1 avgprc
##       <fctr>  <dbl>
## 1       RRNn 214000
## 2       PosA 212500
## 3       PosN 200000
## 4       RRNe 190750
## 5       RRAn 171495
## 6       Norm 166500
## 7       RRAe 142500
## 8      Feedr 140000
## 9     Artery 119550
```

```
house$condition1 <- (house$condition1 == "Artery" | house$condition1 =="Feedr"|
  house$condition1 == "RRAe")*1
house$condition2 <- (house$condition2 == "PosN") * 1

cor(house$garagequal, house$garagecars)
```

```
## [1] 0.5766224
```

```
house %<>% select(-garagequal)
```

```r
fullmodel <- lm(saleprice~.,data = house)
summary(fullmodel)
```

```
##
## Call:
## lm(formula = saleprice ~ ., data = house)
##
## Residuals:
##     Min      1Q  Median
## -188576  -11780     563
##  [ reached getOption("max.print") -- omitted 2 entries ]
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
##  [ reached getOption("max.print") -- omitted 65 rows ]
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 25940 on 1395 degrees of freedom
## Multiple R-squared:  0.8981, Adjusted R-squared:  0.8934
## F-statistic: 192.1 on 64 and 1395 DF,  p-value: < 2.2e-16
```

Checking multicollinearity. Looks good. For the generalized variance inflation factor (normalized by the degree of freedom), everything except one is less than 2.

```r
vif(fullmodel)
```

```
##                     GVIF Df GVIF^(1/(2*Df))
## lotarea         1.433292  1        1.197202
##  [ reached getOption("max.print") -- omitted 29 rows ]
```

Interestingly, `soldminusbuilt` which is `yrsold` - `yearbuilt` becomes insignificant in this smaller model with only the best predictors

```r
house_numeric <- house[,sapply(house,function(x) is.numeric(x))]
house_numeric %<>% select(saleprice, everything())
bestpredictors <- names(house_numeric)[sapply(house_numeric,
function(x) abs(cor(house_numeric$saleprice, x))) >= 0.5][-1]

bestpredictors <- bestpredictors[-6]

bestmodel <- lm(saleprice~overallqual + exterqual + grlivarea +
    kitchenqual + garagecars + neighborhood, data = house)

summary(bestmodel)$r.squared
```

```
## [1] 0.808378
```

Subset with only best predictors

```r
housesubset <- house %>% select(bestpredictors)
```

So, 6 variables capture 0.808378 of the variation in sale price for our model.

Checking assumptions.

```r
cor(housesubset)
```

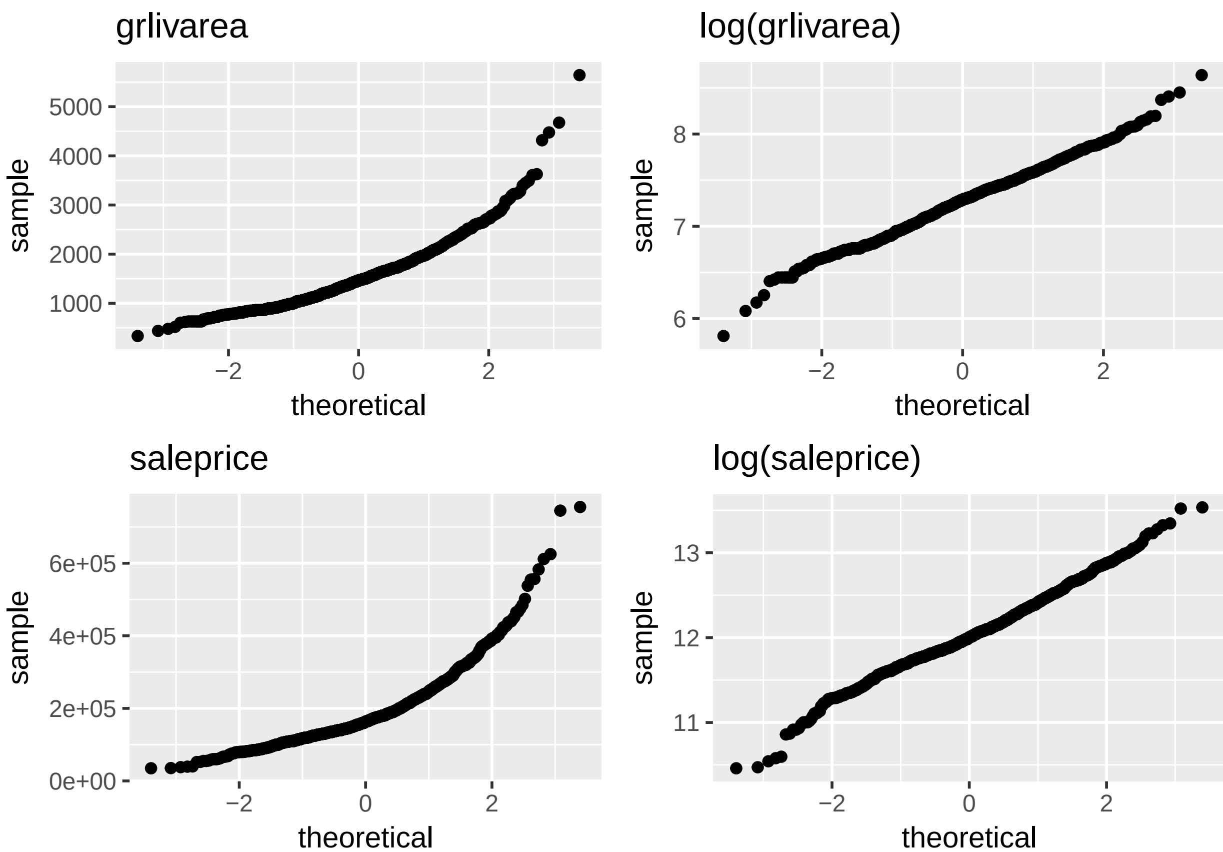
```
##               overallqual exterqual grlivarea kitchenqual garagecars
##  [ reached getOption("max.print") -- omitted 5 rows ]
```

```r
vif(bestmodel)
```

```
##                      GVIF Df GVIF^(1/(2*Df))
## overallqual  3.464742  1        1.861382
##  [ reached getOption("max.print") -- omitted 5 rows ]
```

```r
g1 <- ggplot(housesubset, aes(sample = grlivarea)) +  stat_qq() + ggtitle("grlivarea")
g2 <- ggplot(housesubset, aes(sample = log(grlivarea))) +  stat_qq() + ggtitle("log(grlivarea)")

g3 <- ggplot(house, aes(sample = saleprice)) +  stat_qq() + ggtitle("saleprice")
g4 <- ggplot(house, aes(sample = log(saleprice))) +  stat_qq() + ggtitle("log(saleprice)")
grid.arrange(g1,g2,g3,g4)
```




```r
bestmodel2 <- lm(log(saleprice)~overallqual  + exterqual +  log(grlivarea) +
    kitchenqual + garagecars + neighborhood, data = house)
summary(bestmodel2)
```

```
##
## Call:
## lm(formula = log(saleprice) ~ overallqual + exterqual + log(grlivarea) +
##     kitchenqual + garagecars + neighborhood, data = house)
##
## Residuals:
##      Min       1Q   Median
## -0.97098 -0.07887  0.01184
##  [ reached getOption("max.print") -- omitted 2 entries ]
```

```
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
##  [ reached getOption("max.print") -- omitted 30 rows ]
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1567 on 1430 degrees of freedom
## Multiple R-squared:  0.8492, Adjusted R-squared:  0.8462
## F-statistic: 277.7 on 29 and 1430 DF,  p-value: < 2.2e-16
```

`exterqual` becomes insignificant once we take the log of the response variable

```
bestmodel3 <- lm(log(saleprice)~overallqual  +  log(grlivarea) +
    kitchenqual + garagecars + neighborhood, data = house)
summary(bestmodel3)$r.squared
```

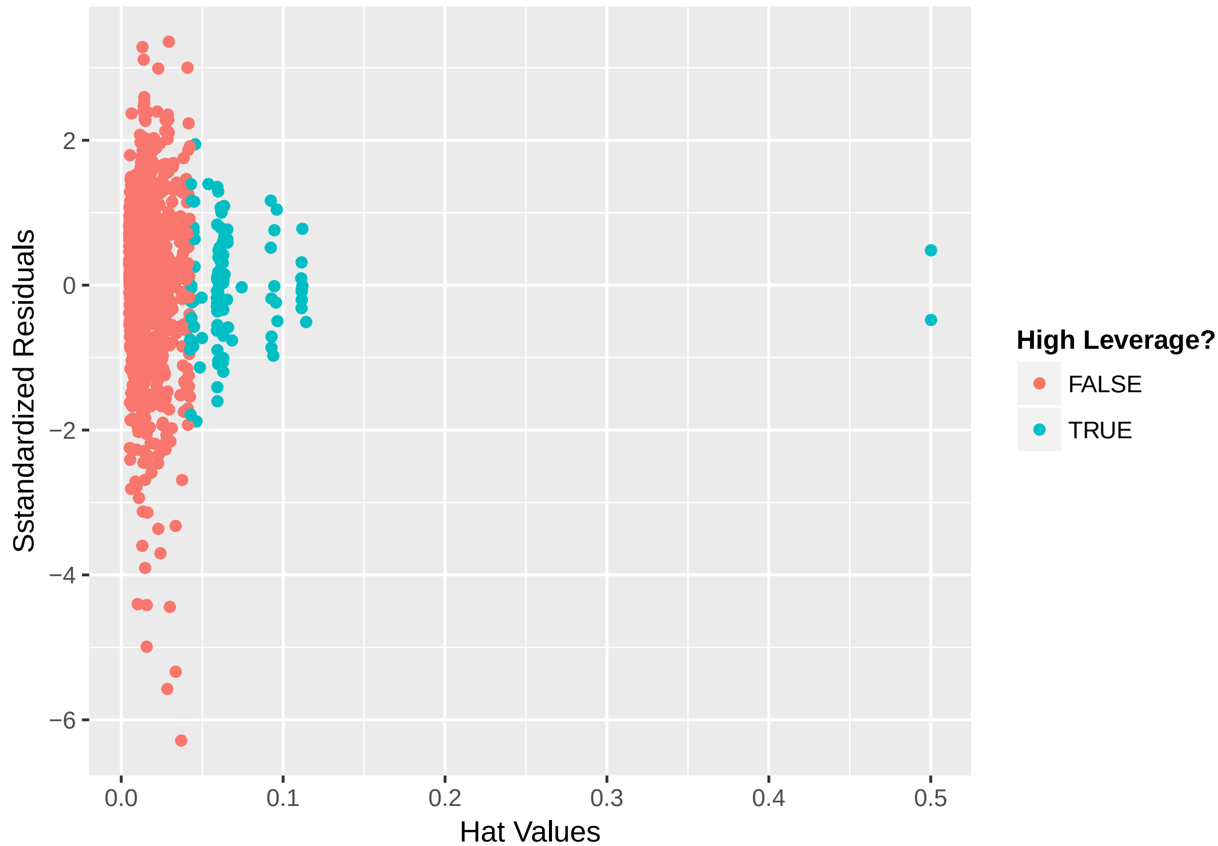
```
## [1] 0.8488445
```

Check for high leverage points. There are 98 high leverage points.

```
( high_leverage <- as.numeric(names(hatvalues(bestmodel3)[(hatvalues(bestmodel3) > 2*ncol(house)/nrow(ho
```

```
## [1]  2 24 54
##  [ reached getOption("max.print") -- omitted 95 entries ]
```

```
lev_df <- data_frame(rstudent = rstudent(bestmodel3),
                     hatvalue = hatvalues(bestmodel3))
lev_df$highlev <- F
lev_df[high_leverage,]$highlev <- T
lev_df %>% ggplot(aes(x=hatvalue, y = rstudent,color = highlev)) + geom_point()+
  xlab("Hat Values") +
  ylab("Sstandardized Residuals") + scale_y_continuous(label=scales::comma) +
  labs(colour = "High Leverage?") +
  theme(legend.title = element_text(size = 10, face = "bold"))
```




```r
length(hatvalues(bestmodel3)[(hatvalues(bestmodel3) > 2*ncol(house)/nrow(house))])
```

```
## [1] 98
```

```r
hatvalues(bestmodel)[hatvalues(bestmodel3) > 0.5]
```

```
##       600       957
## 0.5001289 0.5001289
```

```r
infm <- influence.measures(bestmodel3)
threshhold <- sqrt(2*ncol(house)/nrow(house))
```

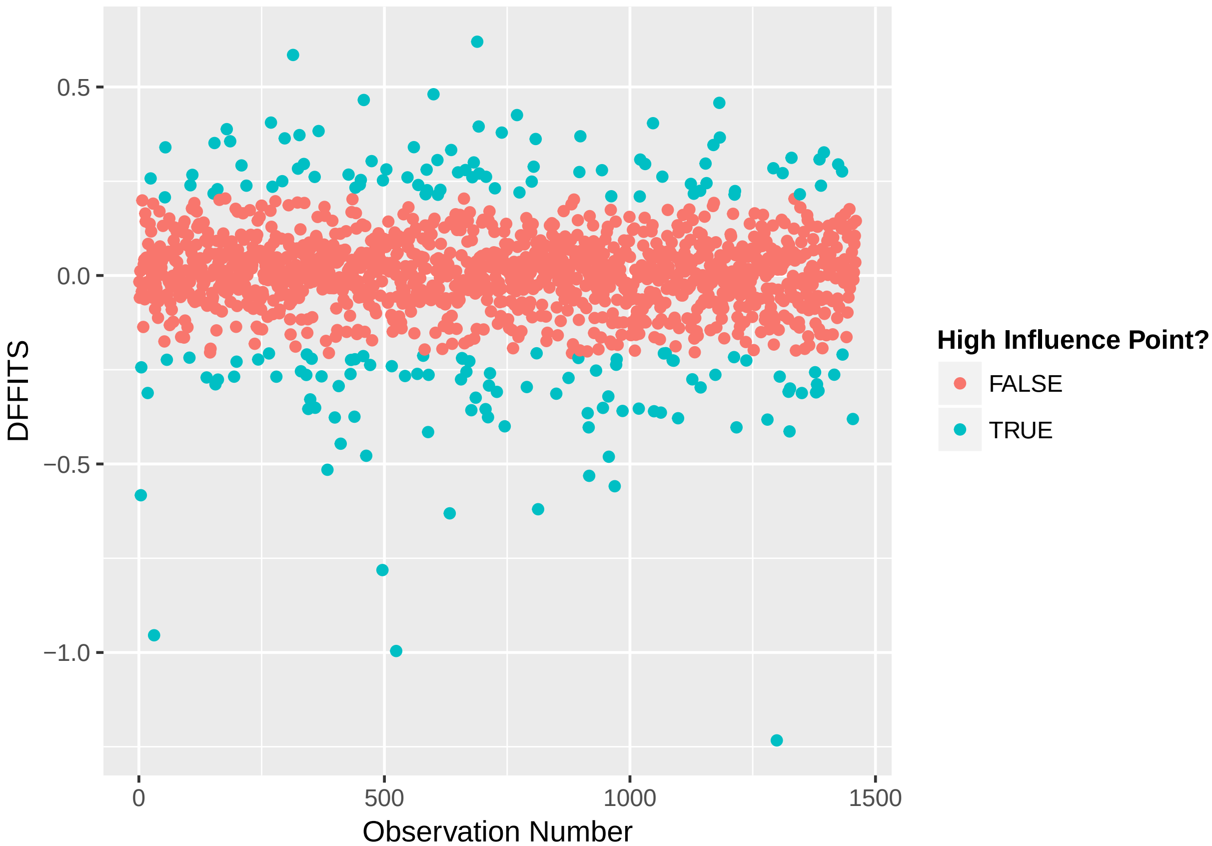
Check for influence points. There are 184 high influence points with a threshhold of $\sqrt{\frac{p}{n}} = 0.2060722$

```r
(high_influence <- which(abs(infm$infmat[,30])>threshhold))
```

```
##  4  5 18
##  4  5 18
##  [ reached getOption("max.print") -- omitted 181 entries ]
```

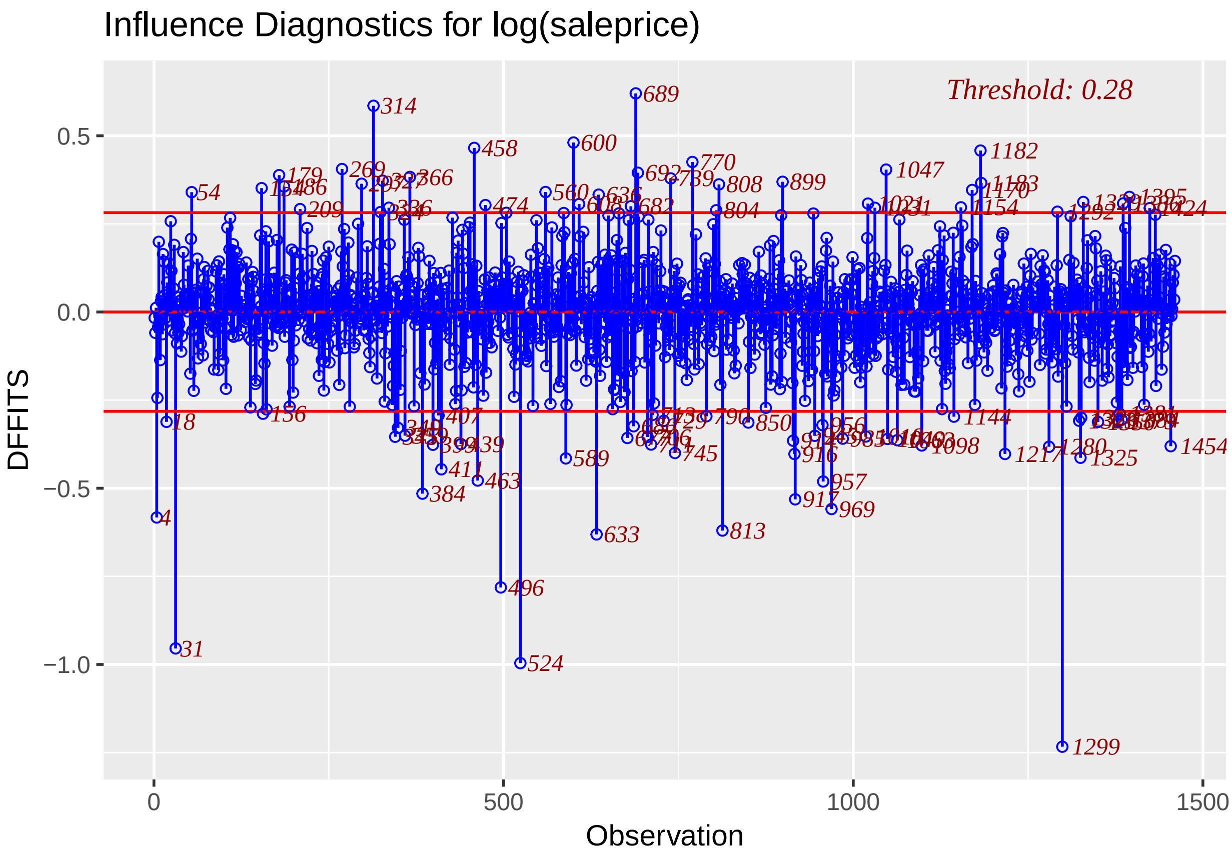
```r
inf_df <- data_frame(dffits = dffits(bestmodel3), index = 1:nrow(house))
inf_df$highinf <- F
inf_df[high_influence,]$highinf <- T

inf_df %>% ggplot(aes(x=index, y=dffits, color = highinf)) + geom_point() +
    xlab("Observation Number") +
  ylab("DFFITS") + scale_y_continuous(label=scales::comma) +
  labs(colour = "High Influence Point?") +
  theme(legend.title = element_text(size = 10, face = "bold"))
```




```r
#install.packages("olsrr")

influence <- ols_dffits_plot(bestmodel3)
```



Let's examine Observation # 1299, and 524

```
house[1299,] %>% View()
house[542,] %>% View()

bestmodel4 <- lm(log(saleprice)~overallqual  +  log(grlivarea) +
    kitchenqual + garagecars + neighborhood, data = house[c(-1299,-542),])
summary(bestmodel4)$r.squared
```

```
## [1] 0.8530995
```

By just removing two points, our Adjusted R-squared went from 0.8458869 to 0.8502211

There are 89 outliers. Let's see what happens if we simply remove the outliers.

```
influenceindex <- unlist(influence$outliers[1])

bestmodelnoinfluence <- lm(log(saleprice)~overallqual  +  log(grlivarea) +
    kitchenqual + garagecars + neighborhood, data = house[-influenceindex,])
summary(bestmodelnoinfluence)$r.squared
```

```
## [1] 0.8889236
```

We see that our Adjusted R-squared went from 0.8502211 to 0.8866905 after removing ALL the influence points.

```
t1 <- names(house)[1:11]
t2 <- names(house)[12:21]
t2[11] <- ""
t3 <- names(house)[22:31]
```

```r
t3[11] <- ""

data_frame(t1,t2,t3) %>%
  knitr::kable(col.names = c("","",""))
```

lotarea	bsmtexposure	garagetype
neighborhood	bsmtfinsf1	garagecars
condition1	bsmtfinsf2	wooddecksf
condition2	bsmtunfsf	poolarea
housestyle	heatingqc	salecondition
overallqual	grlivarea	saleprice
overallcond	bedroomabvgr	remodel
roofmatl	kitchenabvgr	soldminusbuilt
masvnrtype	kitchenqual	summertime
masvnrarea	functional	newtype
exterqual		

```r
house2 <- house
house2[influenceindex, ]$saleprice <- NA
house2$saleprice <- kNN(house2, variable = "saleprice",  k = k)$saleprice
```

```
## Warning in gowerD(don_dist_var, imp_dist_var, weights = weightsx,
## numericalX, : NAs introduced by coercion

## Warning in gowerD(don_dist_var, imp_dist_var, weights = weightsx,
## numericalX, : NAs introduced by coercion

## Warning in gowerD(don_dist_var, imp_dist_var, weights = weightsx,
## numericalX, : NAs introduced by coercion

## Warning in gowerD(don_dist_var, imp_dist_var, weights = weightsx,
## numericalX, : NAs introduced by coercion
```

```r
bestmodelimputeinfluence <- lm(log(saleprice)~overallqual  +  log(grlivarea) +
    kitchenqual + garagecars + neighborhood, data = house2)
summary(bestmodelimputeinfluence)$r.squared
```

```
## [1] 0.866407
```

Let's try our model with all of the relevant variables. First, we notice that the R squared improves by taking the log of `saleprice`, `lotarea`, `grlivarea` and the square root of `bsmtfinsf1`. We also notice that `housestyle` and `masvnrtype` is no longer significant so we remove them.

```r
model31var <- lm(log(saleprice) ~ log(lotarea) +
                   sqrt(bsmtfinsf1)+log(grlivarea)+., data = house)
summary(model31var)$r.squared
```

```
## [1] 0.9255936
```

Accounting for outliers in the full model through imputation

```r
model31varimpute <- lm(log(saleprice) ~ log(lotarea) +
              sqrt(bsmtfinsf1)+log(grlivarea)+., data = house2)
summary(model31varimpute)$r.squared
```

```
## [1] 0.923607
```

We can try removing the outliers, which improved the R squared by a lot. Now, we can test some interaction terms.

```
model31varremove <- lm(log(saleprice) ~ log(lotarea) +
              sqrt(bsmtfinsf1)+log(grlivarea)+., data = house2[-influenceindex,])
summary(model31varremove)$r.squared
```

```
## [1] 0.9469088
```

I remove some variables found to be insignificant.

```
house3 <- house2 %>% select(-condition2,-roofmatl,-garagetype,-poolarea,-remodel)
```

Remove `exterqual`

```
house4 <- house3 %>% select(-exterqual)
```

FINAL MODEL

I test the multicollinearity, significance of variables in the model, normality for our final model.

```
endmodel <- lm(log(saleprice) ~ log(lotarea) +
              sqrt(bsmtfinsf1)+log(grlivarea) +  . -
                lotarea - bsmtfinsf1 - grlivarea,
              data = house4[-influenceindex,])
vifmodel <- lm(log(saleprice) ~ log(lotarea) +
              sqrt(bsmtfinsf1)+log(grlivarea) +  . -
                lotarea - bsmtfinsf1 - grlivarea - neighborhood,
              data = house4[-influenceindex,])
vif(vifmodel) %>% knitr::kable()
```

log(lotarea)	1.428190
sqrt(bsmtfinsf1)	2.789096
log(grlivarea)	4.186157
condition1	1.101531
housestyle	2.174228
overallqual	3.513008
overallcond	1.417131
masvnrtype	2.197963
masvnrarea	2.114535
bsmtexposure	1.145292
bsmtfinsf2	1.171255
bsmtunfsf	3.099577
heatingqc	1.495591
bedroomabvgr	1.792048
kitchenabvgr	1.197538
kitchenqual	2.232100
functional	1.165639
garagecars	2.012814
wooddecksf	1.142861
salecondition	1.850790
soldminusbuilt	2.873251
summertime	1.033354
newtype	2.101804

```r
options(max.print=999)
summary(endmodel)
```

```
##
## Call:
## lm(formula = log(saleprice) ~ log(lotarea) + sqrt(bsmtfinsf1) +
##     log(grlivarea) + . - lotarea - bsmtfinsf1 - grlivarea, data = house4[-influenceindex,
##     ])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.37097 -0.04956  0.00242  0.05213  0.34034
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          6.975e+00  1.090e-01  63.999  < 2e-16 ***
## log(lotarea)         9.534e-02  7.768e-03  12.275  < 2e-16 ***
## sqrt(bsmtfinsf1)     4.981e-03  3.238e-04  15.383  < 2e-16 ***
## log(grlivarea)       4.584e-01  1.581e-02  28.996  < 2e-16 ***
## neighborhoodBrDale  -6.989e-02  3.586e-02  -1.949 0.051482 .
## neighborhoodBrkSide -8.722e-04  2.997e-02  -0.029 0.976786
## neighborhoodClearCr  3.361e-02  3.367e-02   0.998 0.318384
## neighborhoodCollgCr -5.904e-03  2.625e-02  -0.225 0.822088
## neighborhoodCrawfor  1.202e-01  3.051e-02   3.939 8.62e-05 ***
## neighborhoodEdwards -7.077e-02  2.834e-02  -2.497 0.012656 *
## neighborhoodGilbert -9.594e-03  2.788e-02  -0.344 0.730840
## neighborhoodIDOTRR  -8.905e-02  3.337e-02  -2.669 0.007704 **
## neighborhoodMeadowV -7.746e-02  3.486e-02  -2.222 0.026442 *
## neighborhoodMitchel -3.302e-02  2.929e-02  -1.127 0.259840
## neighborhoodNAmes   -2.774e-02  2.737e-02  -1.014 0.310966
## neighborhoodNoRidge  6.854e-02  3.052e-02   2.246 0.024890 *
## neighborhoodNPkVill  3.474e-03  3.923e-02   0.089 0.929444
## neighborhoodNridgHt  8.628e-02  2.731e-02   3.159 0.001619 **
## neighborhoodNWAmes  -3.035e-02  2.847e-02  -1.066 0.286500
## neighborhoodOldTown -7.253e-02  2.933e-02  -2.472 0.013543 *
## neighborhoodSawyer   7.942e-04  2.896e-02   0.027 0.978128
## neighborhoodSawyerW -2.200e-02  2.826e-02  -0.778 0.436435
## neighborhoodSomerst  6.144e-02  2.649e-02   2.319 0.020543 *
## neighborhoodStoneBr  1.185e-01  3.336e-02   3.553 0.000394 ***
## neighborhoodSWISU   -4.328e-02  3.429e-02  -1.262 0.207104
## neighborhoodTimber   2.582e-03  2.973e-02   0.087 0.930826
## neighborhoodVeenker  1.816e-02  4.149e-02   0.438 0.661642
## condition1          -6.237e-02  8.700e-03  -7.169 1.25e-12 ***
## housestyle          -2.119e-02  8.011e-03  -2.646 0.008249 **
## overallqual          5.538e-02  3.509e-03  15.781  < 2e-16 ***
## overallcond          3.705e-02  2.729e-03  13.576  < 2e-16 ***
## masvnrtype          -1.518e-02  7.595e-03  -1.998 0.045887 *
## masvnrarea           5.105e-05  2.120e-05   2.408 0.016185 *
## bsmtexposure         5.005e-02  9.667e-03   5.178 2.59e-07 ***
## bsmtfinsf2           7.966e-05  1.662e-05   4.793 1.83e-06 ***
## bsmtunfsf            6.728e-05  9.757e-06   6.896 8.29e-12 ***
## heatingqc            2.437e-02  6.208e-03   3.926 9.09e-05 ***
## bedroomabvgr        -1.295e-02  4.229e-03  -3.062 0.002246 **
## kitchenabvgr        -5.529e-02  1.282e-02  -4.312 1.74e-05 ***
```

```
## kitchenqual          4.114e-02  5.649e-03   7.283 5.58e-13 ***
## functional           7.798e-02  1.082e-02   7.208 9.50e-13 ***
## garagecars           4.742e-02  4.780e-03   9.921  < 2e-16 ***
## wooddecksf           7.994e-05  2.142e-05   3.731 0.000199 ***
## salecondition        4.426e-02  8.798e-03   5.030 5.57e-07 ***
## soldminusbuilt      -2.112e-03  2.101e-04 -10.050  < 2e-16 ***
## summertime           1.784e-02  4.905e-03   3.638 0.000285 ***
## newtype              1.042e-01  1.323e-02   7.881 6.74e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.08875 on 1324 degrees of freedom
## Multiple R-squared:  0.9439, Adjusted R-squared:  0.9419
## F-statistic:   484 on 46 and 1324 DF,  p-value: < 2.2e-16
```

```r
ks.test(endmodel$residuals, pnorm, mean(endmodel$residuals),
        sd(endmodel$residuals))
```

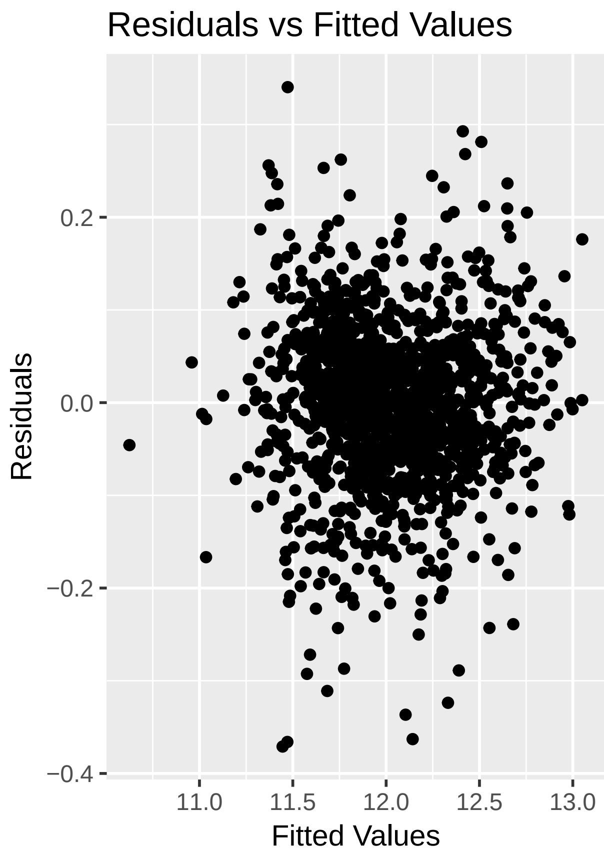
```
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  endmodel$residuals
## D = 0.036643, p-value = 0.05036
## alternative hypothesis: two-sided
```

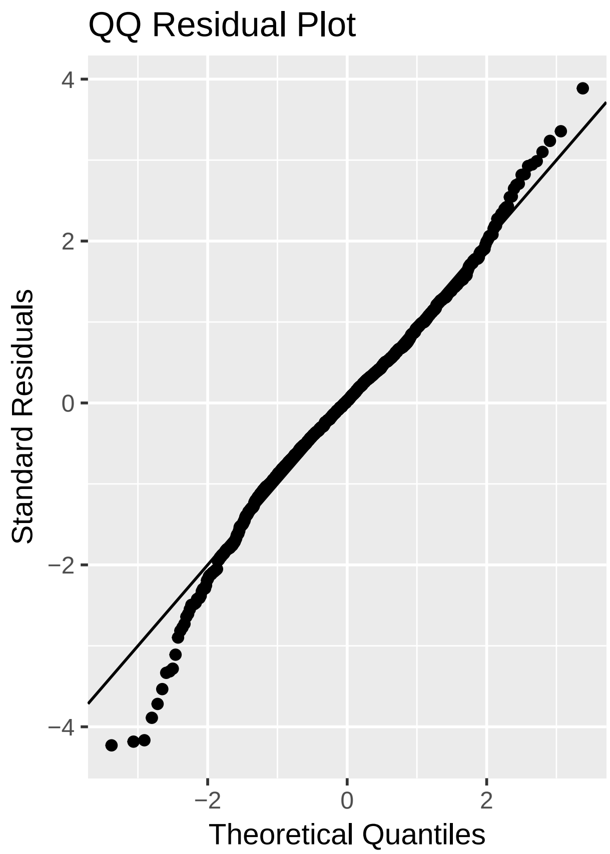
```r
resid_df <- data_frame(res = endmodel$residuals)

r1 <- ggplot(endmodel, aes(.fitted, .resid)) + geom_point() + xlab("Fitted Values") + ylab("Residuals")+
  ggtitle("Residuals vs Fitted Values")

r2 <- ggplot(endmodel, aes(qqnorm(.stdresid)[[1]], .stdresid)) + geom_point(na.rm = T) +geom_abline(inte
  ylab("Standard Residuals") + ggtitle("QQ Residual Plot")

grid.arrange(r1,r2,ncol=2)
```





Checking with LASSO if any variables to remove. Although LASSO recommends to delete `bsmtunsf` and `bedroomabvgr`, removing them lowers the R squared so I will keep them. Many of the neighborhoods are in fact significant so I will leave the non-significant levels in the model anyway.

```
lassorefactor <- function(){

 x <- model.matrix(saleprice ~ ., data = house4)[,-1]
 y <- house$saleprice
 train <- sample(1:nrow(x), nrow(x) / 2)
 test <- (-train)
 y.train <- y[train]
 y.test <- y[test]
 grid.lambda <- 10^seq(10, -2, length = 100)
 lasso.model <- glmnet(x, y, alpha = 1, lambda = grid.lambda)
 set.seed(1)
 cv.out <- cv.glmnet(x[train,], y.train, alpha = 1)
 best.lambda <- cv.out$lambda.min
 lasso.pred <- predict(lasso.model, s = best.lambda, newx = x[test,])
 mspe.lasso <- mean((lasso.pred - y.test)^2)
 final.model <- glmnet(x, y, alpha = 1, lambda = best.lambda)
 c <- coef(final.model)
 ind <- which(c==0)
 variables <- row.names(c)[ind]
 return(variables)
}

lassorefactor()
```

```
## [1] "neighborhoodSawyerW" "bedroomabvgr"
```

Thus, our final model includes the following variables:

```r
names(house4)
```

```
##  [1] "lotarea"        "neighborhood"   "condition1"     "housestyle"
##  [5] "overallqual"    "overallcond"    "masvnrtype"     "masvnrarea"
##  [9] "bsmtexposure"   "bsmtfinsf1"     "bsmtfinsf2"     "bsmtunfsf"
## [13] "heatingqc"      "grlivarea"      "bedroomabvgr"   "kitchenabvgr"
## [17] "kitchenqual"    "functional"     "garagecars"     "wooddecksf"
## [21] "salecondition"  "saleprice"      "soldminusbuilt" "summertime"
## [25] "newtype"
```

```r
signif_var <- house4 %>% select(-neighborhood) %>%
  sapply(function(x) abs(cor(x,house4$saleprice)))

signif_var[signif_var >= 0.5]
```

```
##    overallqual      grlivarea    kitchenqual     garagecars      saleprice
##      0.8131930      0.7019635      0.6832550      0.6635628      1.0000000
## soldminusbuilt
##      0.5646160
```

```r
summary(lm(log(saleprice)~log(grlivarea) +kitchenqual +garagecars + soldminusbuilt + overallqual, data =
```

```
##
## Call:
## lm(formula = log(saleprice) ~ log(grlivarea) + kitchenqual +
##     garagecars + soldminusbuilt + overallqual, data = house4)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.72437 -0.08816  0.00840  0.09265  0.50301
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     8.3094361  0.0983140   84.52   <2e-16 ***
## log(grlivarea)  0.3975474  0.0155102   25.63   <2e-16 ***
## kitchenqual     0.0795455  0.0081666    9.74   <2e-16 ***
## garagecars      0.0741018  0.0070731   10.48   <2e-16 ***
## soldminusbuilt -0.0022851  0.0001722  -13.27   <2e-16 ***
## overallqual     0.0823895  0.0047551   17.33   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1476 on 1454 degrees of freedom
## Multiple R-squared:  0.8389, Adjusted R-squared:  0.8383
## F-statistic:  1514 on 5 and 1454 DF,  p-value: < 2.2e-16
```

Part I: Explanatory Modeling

TASK 1

The five most relevant features that are most relevant in determining a house's sale price are `overallqual`, `grlivarea`, `kitchenqual`, `garagecars`, and `soldminusbuilt`. The fifth variable, `soldminusbuilt` is equal to `yearsold - yearbuilt`.

TASK 2

```
morty<- read_csv("Morty.txt", col_types = cols())
```

```
## Warning: Missing column names filled in: 'X1' [1]
```

Function to transform TEST DATA accordingly. Please run the function transform() and provide the data frame to the argument

```
transform <- function(df){
  names(df) <- tolower(names(df))

  df[is.na(df)] <- "None"
  df$soldminusbuilt <- (df$yrsold - df$yearbuilt)
  df$summertime <- (df$mosold %in% 5:7) * 1
  df$newtype <- (df$saletype == 'New') * 1

  df %<>% select(intersect(names(df), names(house4)))

  df$condition1 <- (df$condition1 == "Artery" |
      df$condition1 =="Feedr"| df$condition1 == "RRAe")*1

  df$housestyle <- (df$housestyle == "2Story" |
                    df$housestyle == "2.5Fin")*1

  df$masvnrtype <- (df$masvnrtype != "None") * 1
  df$bsmtexposure <- (df$bsmtexposure == "Gd") * 1

  df$heatingqc <- as.numeric(factor(df$heatingqc,
  levels = c("Po","Fa","TA","Gd","Ex"), labels = 1:5))

  df$kitchenqual <-  as.numeric(factor(df$kitchenqual,
    levels = c("Po","Fa","TA","Gd","Ex"), labels = 1:5))

  df$functional <- (df$functional == "Typ") * 1
  df$salecondition <- (df$salecondition == "Normal") * 1
  return(df)
}
morty2 <- transform(morty)
```

`morty2` is our transformed data. Note that it only has 25 variables

```r
confmorty <- exp(predict(endmodel, morty2, interval = "confidence", level = 0.95))
confmorty %>% knitr::kable()
```

fit	lwr	upr
184739.7	173389	196833.6

```r
morty_stat <- as.numeric(unlist(morty2))
```

```
## Warning: NAs introduced by coercion
```

```r
names(morty_stat) <- names(morty2)
mean_stat <- sapply(house4, function(x) round(mean(x)))
```

```
## Warning in mean.default(x): argument is not numeric or logical: returning
## NA
```

```r
morty_stat
```

```
##        lotarea   neighborhood     condition1     housestyle    overallqual
##          14115             NA              0              0              5
##    overallcond     masvnrtype     masvnrarea    bsmtexposure    bsmtfinsf1
##              5              0              0              0            732
##     bsmtfinsf2      bsmtunfsf      heatingqc      grlivarea    bedroomabvgr
##              0             64              5           1362              1
##    kitchenabvgr    kitchenqual     functional     garagecars     wooddecksf
##              1              3              1              2             40
##  salecondition      saleprice soldminusbuilt     summertime        newtype
##              1         143000             16              0              0
```

```r
mean_stat
```

```
##        lotarea   neighborhood     condition1     housestyle    overallqual
##          10517             NA              0              0              6
##    overallcond     masvnrtype     masvnrarea    bsmtexposure    bsmtfinsf1
##              6              0            103              0            444
##     bsmtfinsf2      bsmtunfsf      heatingqc      grlivarea    bedroomabvgr
##             47            567              1           1515              3
##    kitchenabvgr    kitchenqual     functional     garagecars     wooddecksf
##              1              4              1              2             94
##  salecondition      saleprice soldminusbuilt     summertime        newtype
##              1         179378             37              0              0
```

```r
(improve <- house4 %>% select(-neighborhood,-saleprice, -soldminusbuilt) %>%  sapply(function(x) abs(co
```

```
##    overallqual      grlivarea    kitchenqual     garagecars     masvnrarea
##     0.81319300     0.70196347     0.68325498     0.66356282     0.48261046
##      heatingqc     masvnrtype     bsmtfinsf1        newtype     wooddecksf
##     0.45069412     0.40311867     0.38828646     0.38105085     0.31656892
##   bsmtexposure     housestyle      bsmtunfsf        lotarea     condition1
##     0.28015224     0.26116978     0.23805450     0.20966584     0.18606670
##   bedroomabvgr  salecondition   kitchenabvgr     functional    overallcond
##     0.16361002     0.15844047     0.14065425     0.12616236     0.10969565
##     summertime     bsmtfinsf2
##     0.03825775     0.02125635
```

```r
improve %>% knitr::kable()
```

overallqual	0.8131930
grlivarea	0.7019635
kitchenqual	0.6832550
garagecars	0.6635628
masvnrarea	0.4826105
heatingqc	0.4506941
masvnrtype	0.4031187
bsmtfinsf1	0.3882865
newtype	0.3810509
wooddecksf	0.3165689
bsmtexposure	0.2801522
housestyle	0.2611698
bsmtunfsf	0.2380545
lotarea	0.2096658
condition1	0.1860667
bedroomabvgr	0.1636100
salecondition	0.1584405
kitchenabvgr	0.1406542
functional	0.1261624
overallcond	0.1096957
summertime	0.0382578
bsmtfinsf2	0.0212564

`overallqual` and `kitchenqual` are in the top 3 for correlation with saleprice. `grlivarea` is difficult/nearly impossible to improve so we will move on to the next variable. `masvnrarea` and `heatingqc` are fairly close. We see that Morty already has the highest `heatingqc` possible so `masvnrarea` should be considered.

Conclusion: Morty should try to improve the `overallqual`, which is the overall material and finish of the house. This may mean repainting some areas on the house to make it look nicer. Morty currently has a rating of 5 out 10 (average rating is 6 out of 10) so there is definitely room for improvement. Next, Morty should improve `kitchenqual`, which is kitchen quality. Maybe, there can be some remodeling done or fixing anything that is either old, or possibly broken. Morty has a rating of 3 out of 5 compared to the average rating of 4 out of 5. Finally, he can increase `masvnrarea`. He currently does not have a masonary veneer so he can consider building one because he might be able to make a profit from it.

We believe that Morty can sell his house for a *maximum* of 196,833.6. The 95 % confidence interval goes from 184,739.7 to 196,833.6 with an average of 173,389.

Part II Predictive Modeling

Ordinary Least Squares

```r
set.seed(1)
train <- sample(nrow(house)*.8)
test <- (-train)
housetrain <- house4[train,]
housetest <- house4[test,]

OLS_train <- lm(log(saleprice) ~ log(lotarea) +
              sqrt(bsmtfinsf1)+log(grlivarea) +  . -
```

```
                lotarea - bsmtfinsf1 - grlivarea,
              data = housetrain[-influenceindex,])
OLS_predict <- exp(predict(OLS_train, housetest,
      interval = "prediction", level = 0.95, type = "response"))
prettyNum(mean((OLS_predict[,1] - housetrain$saleprice)^2), big.mark = ",")
```

```
## [1] "10,670,105,697"
```

```
GLS_train <- glm(log(saleprice) ~ log(lotarea) +
              sqrt(bsmtfinsf1)+log(grlivarea) +  . -
                lotarea - bsmtfinsf1 - grlivarea,
              data = housetrain[-influenceindex,])
GLS_predict <- exp(predict(OLS_train, housetest,
      interval = "prediction", level = 0.95, type = "response"))
prettyNum(mean((GLS_predict[,1] - housetrain$saleprice)^2), big.mark = ",")
```

```
## [1] "10,670,105,697"
```

Define the function to generate models for ridge, lasso and elastic net

```
model_func <- function(input_data, input_alpha){
set.seed(1)
x <- model.matrix(saleprice ~ ., data = input_data)[,-1]
y <- house$saleprice
train <- sample(nrow(house)*.8)
test <- (-train)
y.train <- y[train]
y.test <- y[test]
grid.lambda <- 10^seq(10, -2, length = 100)
model.train <- glmnet(x[train, ], y.train, alpha = input_alpha, lambda = grid.lambda)
set.seed(1)
cv.out <- cv.glmnet(x[train,], y.train, alpha = input_alpha)
best.lambda <- cv.out$lambda.min
pred <- predict(model.train, s = best.lambda, newx = x[test,])
mspe <- mean((pred - y.test)^2)
final.model <- glmnet(x, y, alpha = input_alpha, lambda = best.lambda)
c <- coef(final.model)
return(c(mspe, final.model))
}
```

Ridge regression model, λ set at 0

```
ridge_result <- model_func(house4,0)
ridge_mspe <- ridge_result[1]
prettyNum(ridge_mspe, big.mark = ",")
```

```
##
## "1,769,352,685"
```

lasso regression model, lambda set at 1

```
lasso_result <- model_func(house4,1)
lasso_mspe <- lasso_result[1]
prettyNum(lasso_mspe, big.mark = ",")
```

```
##
## "1,880,105,933"
```

elastic net regression, lambda set at 0.5

```r
elastic_result <- model_func(house4,0.5)
elastic_mspe <- elastic_result[1]
prettyNum(elastic_mspe, big.mark = ",")
```

```
##
## "1,875,864,966"
```

λ is chosen to determine whether we are performing Ridge ($\lambda = 0$), Lasso ($\lambda = 1$), Elastic Net ($\lambda = 0.5$). The tuning parameters in the respective models is chosen via cross validation after trying 100 different ones.

```r
help(cv.glmnet)
```

Justification

Our ridge model performed the best and has the lowest MSPE. This makes sense, given that our data is very sparse, containing many zeros.

```r
countzero <- function(x){
  sum(x==0)
}
sapply(house4, function(x) countzero(x))
```

```
##        lotarea   neighborhood     condition1     housestyle    overallqual
##              0              0           1320           1007              0
##    overallcond     masvnrtype     masvnrarea   bsmtexposure     bsmtfinsf1
##              0            872            869           1326            467
##     bsmtfinsf2      bsmtunfsf      heatingqc      grlivarea   bedroomabvgr
##           1293            118            719              0              6
##    kitchenabvgr    kitchenqual     functional     garagecars     wooddecksf
##              1              0            100             81            761
##  salecondition      saleprice soldminusbuilt     summertime        newtype
##            262              0             64            769           1338
```

Many of these are boolean variables, but we can see that `masvnrarea`, `bsmtfinsf`, `bsmtfinsf2`, and `bsmtunsf` all have zeros. We chose all of these variables because we found them to be statistically significant in our model.

```r
house4 %>% select(-neighborhood) %>% sapply(function(x) abs(cor(x, house4$saleprice))) %>% sort(decreas
```

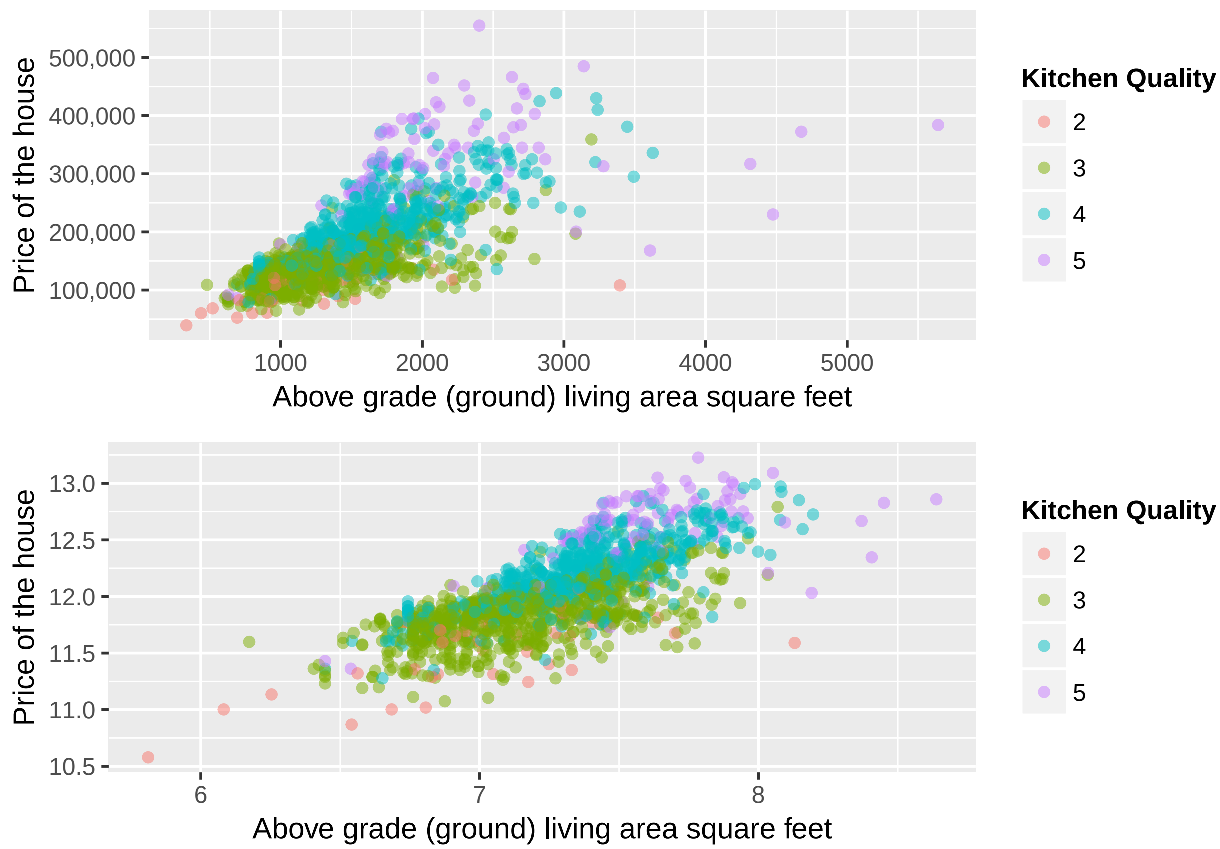
```
##      saleprice    overallqual      grlivarea    kitchenqual     garagecars
##     1.00000000     0.81319300     0.70196347     0.68325498     0.66356282
## soldminusbuilt     masvnrarea      heatingqc     masvnrtype     bsmtfinsf1
##     0.56461597     0.48261046     0.45069412     0.40311867     0.38828646
##        newtype     wooddecksf   bsmtexposure     housestyle      bsmtunfsf
##     0.38105085     0.31656892     0.28015224     0.26116978     0.23805450
##        lotarea     condition1   bedroomabvgr  salecondition   kitchenabvgr
##     0.20966584     0.18606670     0.16361002     0.15844047     0.14065425
##     functional    overallcond     summertime     bsmtfinsf2
##     0.12616236     0.10969565     0.03825775     0.02125635
```

Some variables have more impact than others but nevertheless they are statistically significant in our model so we keep them. Three of these variables are generated from other variables. We created `summertime` partly because of common sense and after plotting the distribution of houses being sold by month, we saw a peak in the summer months. This makes sense practically because people tend to have more time during the summer and thus are more likely to buy a house. Secondly, we created `soldminusbuilt` because we felt that the difference between `yearsold` and `yearbuilt` is more useful together rather than seperately. The third variable we created is a boolean for `saletype` to indicate a house that was "just constructed and sold", which from a common sense perspective, can make the house go much higher. Many of the variables are condensed into smaller levels. Many levels have very few observations so we feel they are not significant enough to have their own level. This helps to prevent overfitting when predicting new values. We chose to not have too many variables in our model to also prevent overfitting. We confirmed the validity of our variables through LASSO regression. Lasso didn't really eliminate any variables, which supports the statistical significance of our predictors.

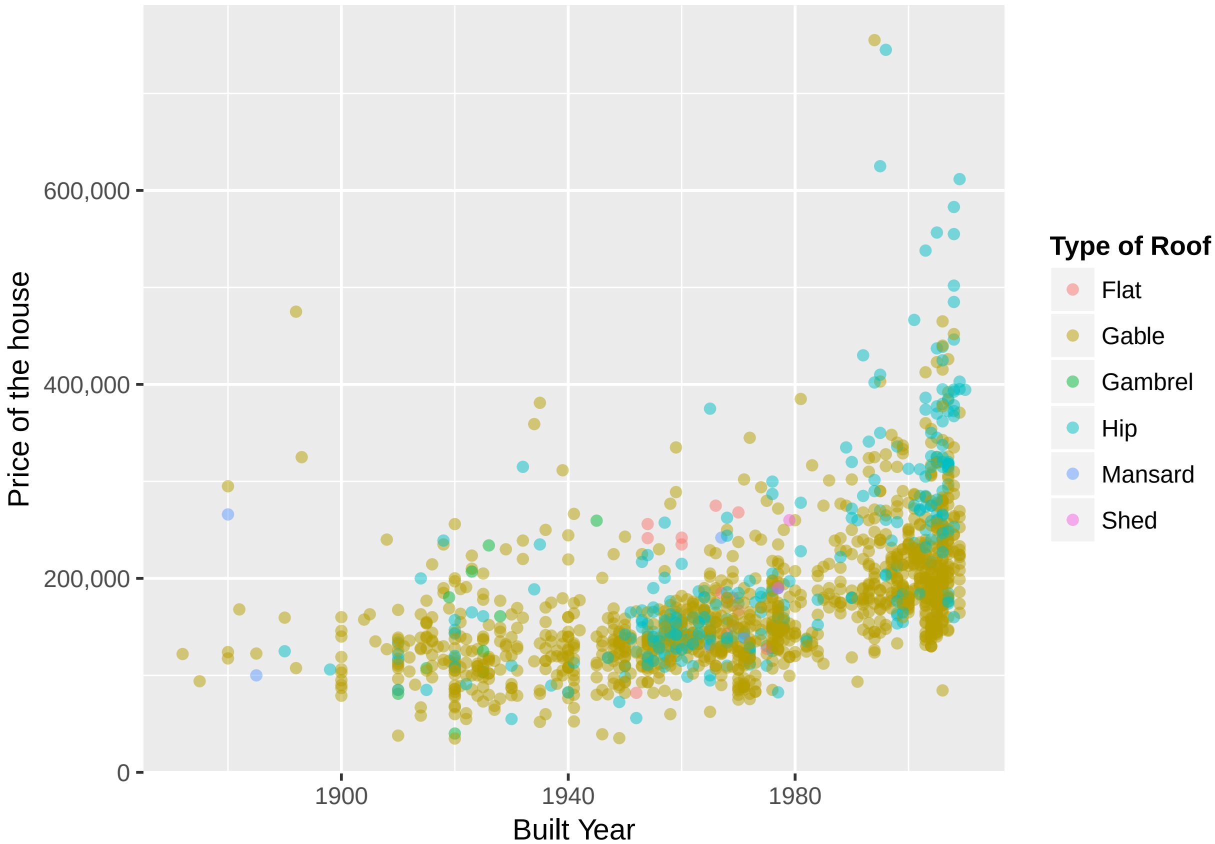
Exploratory Data Analysis

```
p1 <- house4 %>% ggplot(aes(x=grlivarea, y = saleprice,
                         color = factor(kitchenqual))) + geom_point(alpha = 0.5) +
  xlab("Above grade (ground) living area square feet") +
  ylab("Price of the house") + scale_y_continuous(label=scales::comma) +
  labs(colour = "Kitchen Quality") +
  theme(legend.title = element_text(size = 10, face = "bold"))
p2 <- house4 %>% ggplot(aes(x=log(grlivarea), y = log(saleprice),
                         color = factor(kitchenqual))) + geom_point(alpha = 0.5) +
  xlab("Above grade (ground) living area square feet") +
  ylab("Price of the house") + scale_y_continuous(label=scales::comma) +
  labs(colour = "Kitchen Quality") +
  theme(legend.title = element_text(size = 10, face = "bold"))

grid.arrange(p1,p2,ncol=1)
```



```
house0%>% ggplot(aes(x=yearbuilt, y = saleprice,
                     color = factor(roofstyle))) + geom_point(alpha = 0.5) +
  xlab("Built Year") +
  ylab("Price of the house") + scale_y_continuous(label=scales::comma) +
  labs(colour = "Type of Roof") +
  theme(legend.title = element_text(size = 10, face = "bold"))
```




```
ggsave("plot2.png")
```

```
## Saving 6.5 x 4.5 in image
```

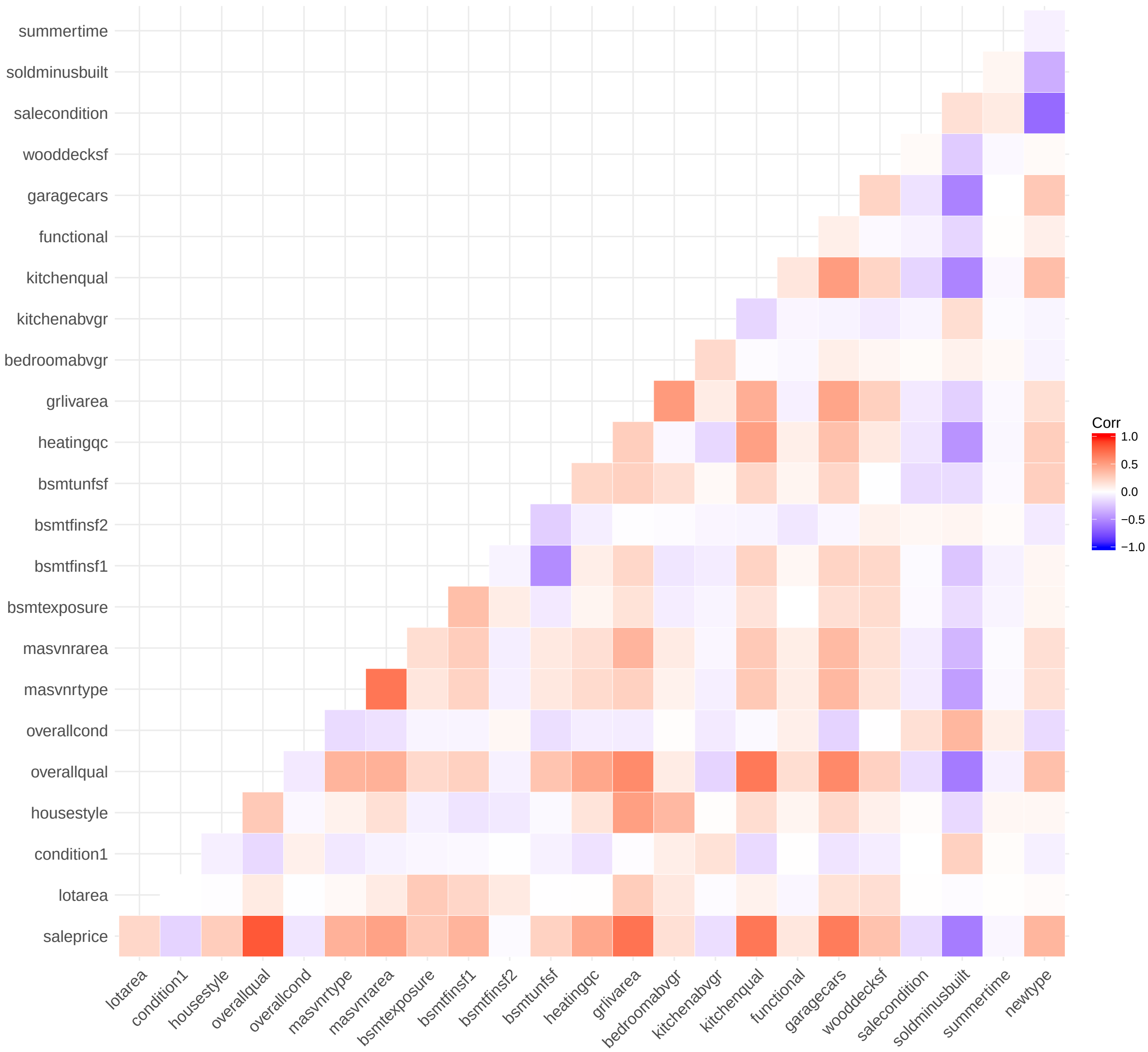
Getting all of the numeric variables.

```
house_numeric <- house4[,sapply(house4,function(x) is.numeric(x))]

house_numeric %<>% select(saleprice, everything())
#install.packages("ggcorrplot")

cor_matrix <- cor(house_numeric)

ggcorrplot(cor_matrix, type = "lower", outline.col = "white", insig = "blank")
```


summertime
soldminusbuilt
salecondition
wooddecksf
garagecars
functional
kitchenqual
kitchenabvgr
bedroomabvgr
grlivarea
heatingqc
bsmtunfsf
bsmtfinsf2
bsmtfinsf1
bsmtexposure
masvnrarea
masvnrtype
overallcond
overallqual
housestyle
condition1
lotarea
saleprice
lotarea
condition1
housestyle
overallqual
overallcond
masvnrtype
masvnrarea
bsmtexposure
bsmtfinsf1
bsmtfinsf2
bsmtunfsf
heatingqc
grlivarea
bedroomabvgr
kitchenabvgr
kitchenqual
functional
garagecars
wooddecksf
salecondition
soldminusbuilt
summertime
newtype
Corr
1.0
0.5
0.0
−0.5
−1.0