Replication Presentation

Applied Stats II

Due: April 11, 2022

Incorporating Multiple Linear Regression in Predicting the House Prices Using a Big Real Estate Dataset with 80 Independent Variables

The study that was selected for this replication assignment did not have an available R file, so I wrote the code based on the methodology in the paper. This document will go through the steps I took and which parts of the study I altered.

1. Prepossessing

Dealing with NA's

Firstly, I looked at the percent of NA's, the following code showed that 5.8% of the dataset were NA's.

```
# calculating the product of dimensions of dataframe
totalcells = prod(dim(data))
# calculating the number of cells with na
missingcells = sum(is.na(data))
# calculating percentage of missing values
percentage = (missingcells * 100 )/(totalcells)
print (percentage)
```

The first issue I noticed with the study was that they explained that they replaced all the NA's with the mean of the column, however it seems as though they were treating all the variables as numerical. To get rid of all the NA's I firstly removed the columns with over 1000 NA's, like the study and then in the numerical columns, replaced the NA's with the mean and in the categorical columns, the NA with the mode. The following code was used:

```
missing_values <- lapply(data,function(x) { length(which(is.na(x)))})
missing_values
#$Pool.QC
#[1] 2917
#$Misc.Feature
#[1] 2824
#$Fence
#[1] 2358
```

```
#$Fireplace.Qu
 #[1] 1422
#$Alley
 #[1] 2732

#Remove these features from the dataset
drop <- c("Pool.QC","Misc.Feature", "Fence", "Fireplace.Qu", "Alley")
data = data[,!(names(data) %in% drop)]

#####################################################
#to replace NA's in data with means of the column.
#####################################################

#The Mean for Numeric Data
for(i in 1:ncol(data)){
  data[is.na(data[,i]), i] <- mean(data[,i], na.rm = TRUE)
}

#Below is for Categorical data

calc_mode <- function(x){
  distinct_values <- unique(x)
  distinct_tabulate <- tabulate(match(x, distinct_values))
  distinct_values[which.max(distinct_tabulate)]
}

#Need to put in the mode for the categorical data

data$Garage.Qual[is.na(data$Garage.Qual)]<-calc_mode(data$Garage.Qual)
data$Garage.Cond[is.na(data$Garage.Cond)]<-calc_mode(data$Garage.Cond)
data$Garage.Finish[is.na(data$Garage.Finish)]<-calc_mode(data$Garage.Finish)
data$Garage.Type[is.na(data$Garage.Type)]<-calc_mode(data$Garage.Type)
data$BsmtFin.Type.2[is.na(data$BsmtFin.Type.2)]<-calc_mode(data$BsmtFin.Type
    .2)
data$BsmtFin.Type.1[is.na(data$BsmtFin.Type.1)]<-calc_mode(data$BsmtFin.Type
    .1)
data$Bsmt.Exposure[is.na(data$Bsmt.Exposure)]<-calc_mode(data$Bsmt.Exposure)
data$Bsmt.Cond[is.na(data$Bsmt.Cond)]<-calc_mode(data$Bsmt.Cond)
data$Bsmt.Qual[is.na(data$Bsmt.Qual)]<-calc_mode(data$Bsmt.Qual)

#Now there are no NA's in the data
sum(is.na(data))
```

Now that there were no NA's in the data I conducted exploratory data analysis, as was done in the study.

2. Descriptive Visualisations

Firstly examining the distribution of the outcome variable "SalePrice"

```
ggplot(data = data, aes(x=SalePrice)) + geom_histogram(color="blue", fill= "
    blue", bins = 70)
```

This produced the following plot:

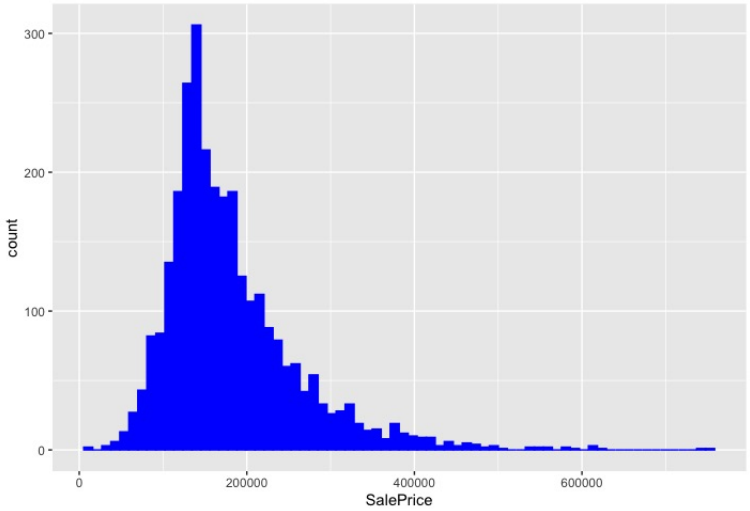

Figure 1: Histogram of the Outcome Variable: SalePrice

The skew to the left is visible and so I followed the steps in the study and log transformed the varable.

```
data$SalePrice <- log(data$SalePrice)
ggplot(data = data, aes(x=SalePrice)) + geom_histogram(color="blue", fill= "
    blue", bins = 70)
```

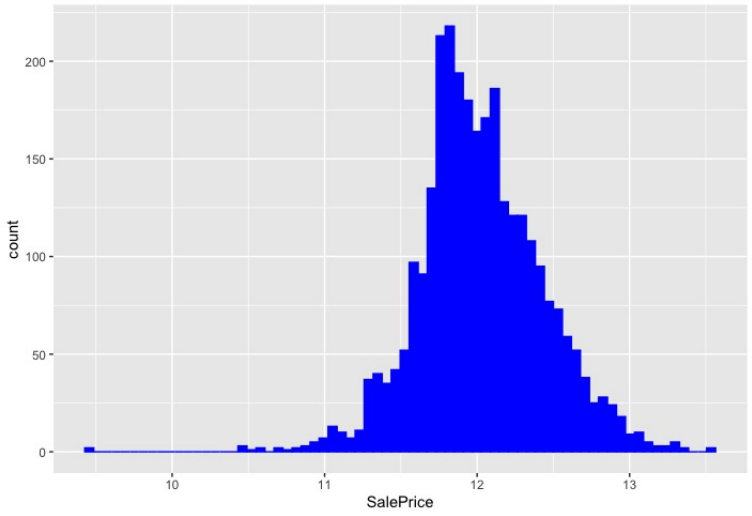

Figure 2: Histogram of the Outcome Variable: Log(SalePrice)

The next visualisation is for the Neighborhood category. I recreated the boxplot from the study below:

```
data %>%
  mutate(Neighborhood = fct_reorder(Neighborhood, SalePrice)) %>%
  ggplot( aes(x=Neighborhood, y=SalePrice, fill=Neighborhood)) +
  geom_boxplot() +
  xlab("") +
  theme_bw()
```

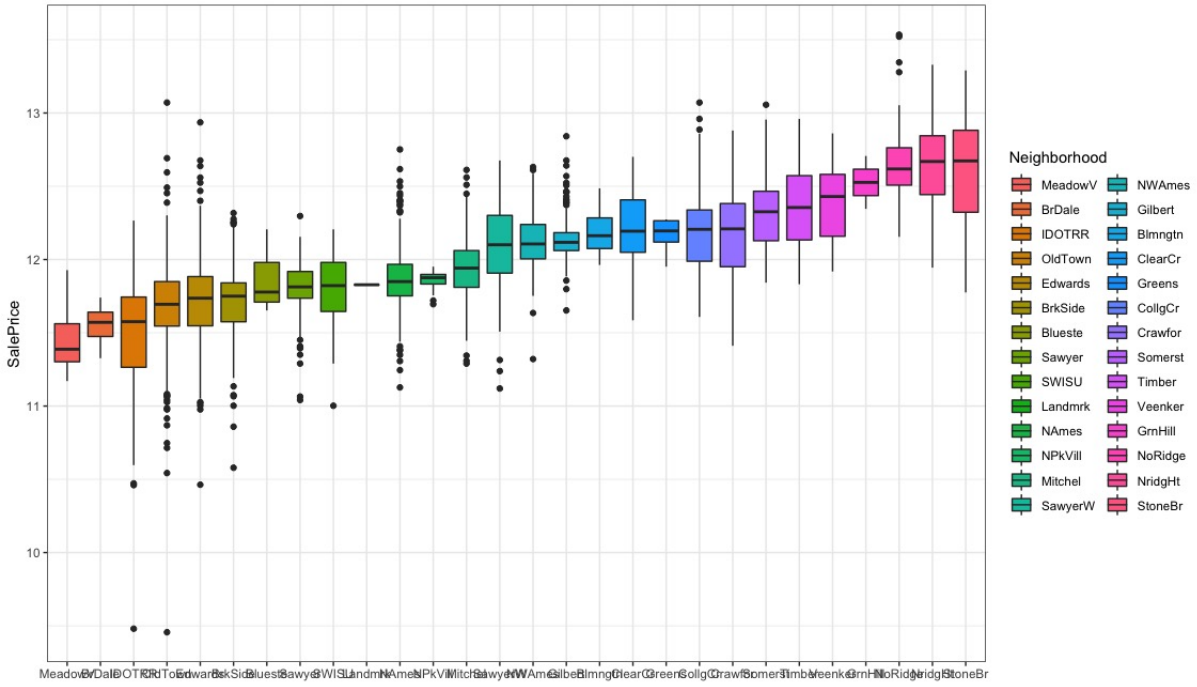

Figure 3: Boxplot of the Variable: Neighborhood

Next the correlations were looked at. The study emphasized the use of understanding if variables were highly correlated, the author stated:

"Multicollinearity assumption: linear regression model assumes that there is little or no multicollinearity in the data. Multicollinearity occurs when the independent variables are highly correlated with each other. We have checked the correlation between all predictors. Some predictors were found to be highly correlated with each other and with the response variable SalePrice. We have identified these highly correlated predictors and removed the predictors with more than 50% correlation from the dataset in Step 10. So, this assumption holds.

Therefore it was very important to replicate what they did. The following code was used to produced a correlation plot, showing correlations that were above 0.5. (As there were 80 variables in the dataset, I set this limit so that it could be legible)

```
library(corrplot)
corr_simple <- function(data=data,sig=0.5){A
  df_cor <- data %>% mutate_if(is.character, as.factor)
  df_cor <- df_cor %>% mutate_if(is.factor, as.numeric)
  corr <- cor(df_cor)
  corr[lower.tri(corr,diag=TRUE)] <- NA
  corr[corr == 1] <- NA
  corr <- as.data.frame(as.table(corr))
  corr <- na.omit(corr)
  corr <- subset(corr, abs(Freq) > sig)
```

```
  corr <- corr[order(-abs(corr$Freq)),]
  print(corr)
  mtx_corr <- reshape2::acast(corr, Var1~Var2, value.var="Freq")

  corrplot(mtx_corr, is.corr=FALSE, tl.col="black", na.label=" ")
}
corr_simple(data)
```

This produced the following plot, which confirms the high correlations which were highlighted in the study:

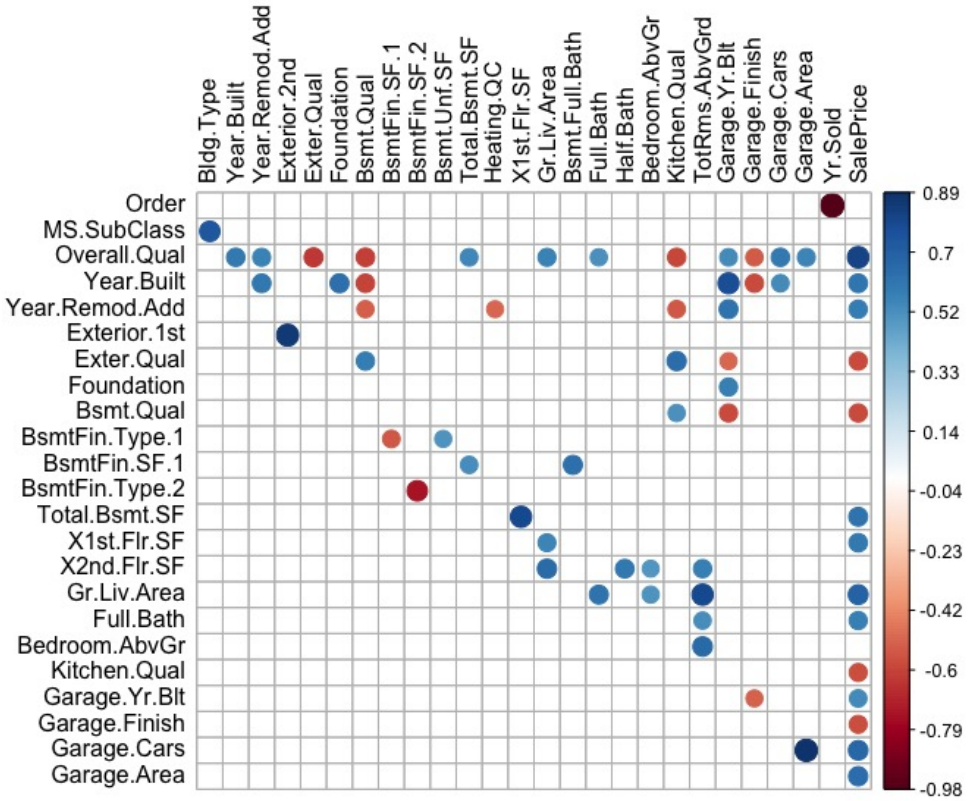

Figure 4: Plot of Correlations over 0.5

Next the features that had near zero variance were removed:

```
library(caret)
names(data)[nearZeroVar(data)]
```

Checking Linearity

```
ggplot(data, aes(x=Overall.Qual, y=SalePrice)) +
  geom_point()+
  geom_smooth(method=lm)

ggplot(data, aes(x=Total.Bsmt.SF, y=SalePrice)) +
  geom_point()+
  geom_smooth(method=lm)
```

The following plots show that the data fits the linearity assumption

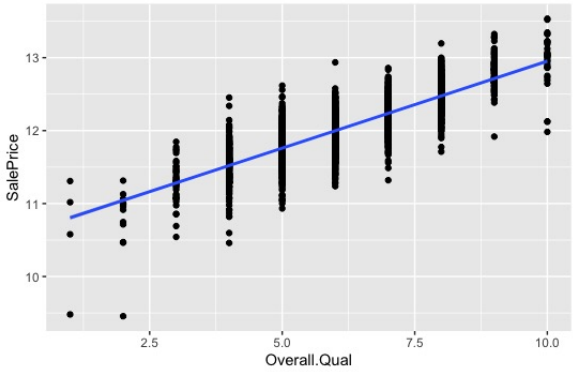

Figure 5: Linearity Overall Quality

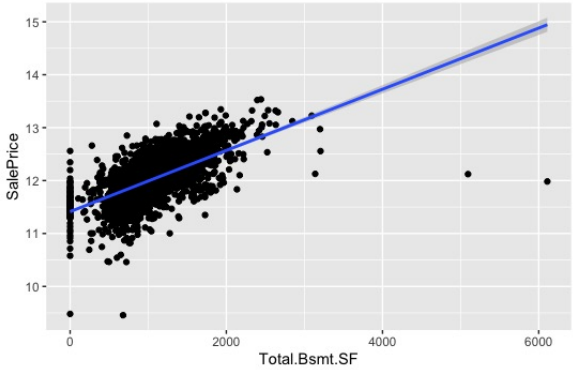

Figure 6: Linearity BSMT SF

3. Random Forest for Feature Selection

By looking through the study, it seemed as though, despite the previous prepossessing steps, the random forest was ran on the original data. I therefore ran the random forest on the original data to explore whether this was the case.

```
library(randomForest)

set.seed(100)
train <- sample(nrow(data), 0.5*nrow(data), replace = FALSE)
TrainSet <- data[train,]
ValidSet <- data[-train,]
summary(TrainSet)
summary(ValidSet)

RF_model <- randomForest(SalePrice ~ ., data = TrainSet, importance = TRUE)
RF_model

importance(RF_model)
varImpPlot(RF_model)
```

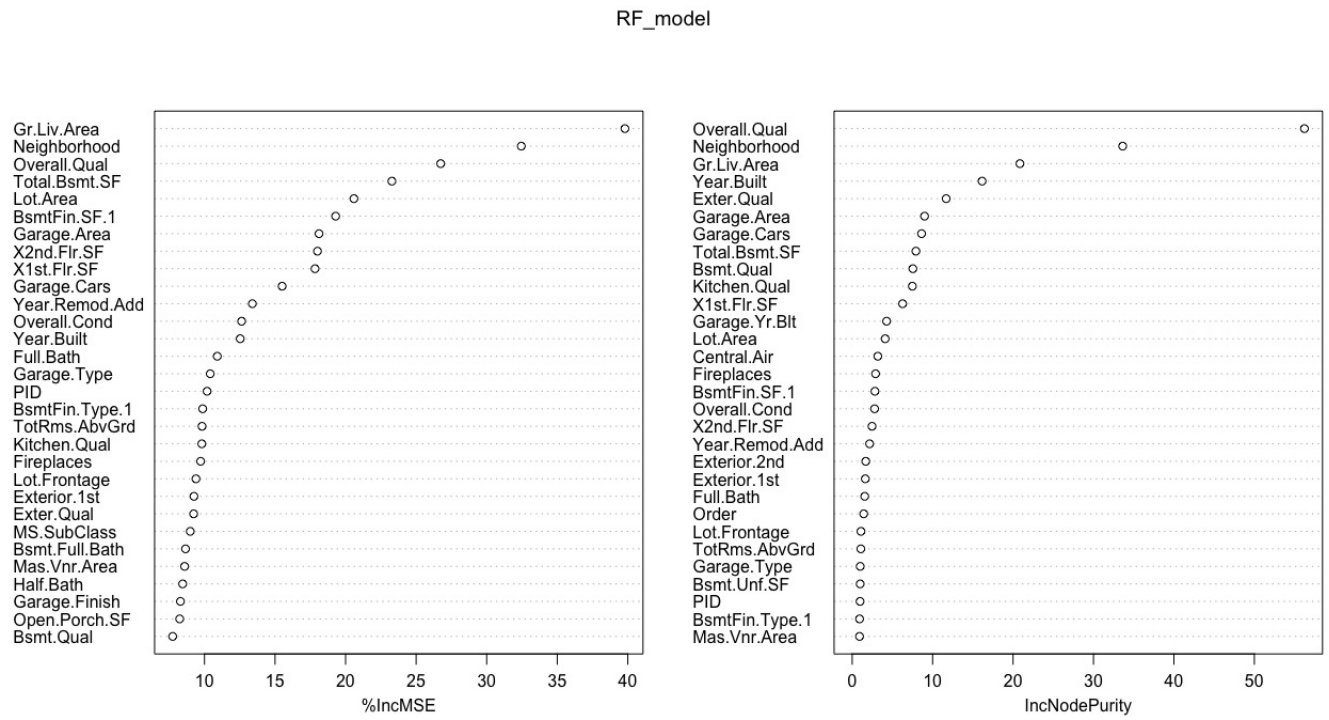

Figure 7: Random Forest on the Original Data

The above plot shows that the most important features that were selected for the regression were from the random forest ran on the orginal data, not on the data with out highly correlated variables or near zero variance.

I then ran the random forest on the data which did take into account the prepossessing. Firstly I removed these items from the dataset:

```
drop <- c("Overall.Qual",
          "Gr.Liv.Area",
          "Garage.Cars",
          "Garage.Area",
          "Total.Bsmt.Sf",
          "X1st.Flr.SF",
          "Full.Bath",
          "Tot.Rms.Abv.Grd",
          "Year.Built",
          "Year.Remod.Add",
          "BsmtFin.SF.1",
          "X1st.Flr.SF",
          "Garage.Cars",
          "X2nd.Flr.SF",
          "Bsmt.Unf.SF",
          "Central.Air",
          "Bedroom.AbvGr",
          "Garage.Finish")

data = data[,!(names(data) %in% drop)]
```

I then re-ran the Random forest

```
set.seed(100)
train <- sample(nrow(data), 0.5*nrow(data), replace = FALSE)
TrainSet <- data[train,]
ValidSet <- data[-train,]
summary(TrainSet)
summary(ValidSet)

RF_model2 <- randomForest(SalePrice ~ ., data = TrainSet, importance = TRUE)
RF_model2

importance(RF_model2)
varImpPlot(RF_model2)
```

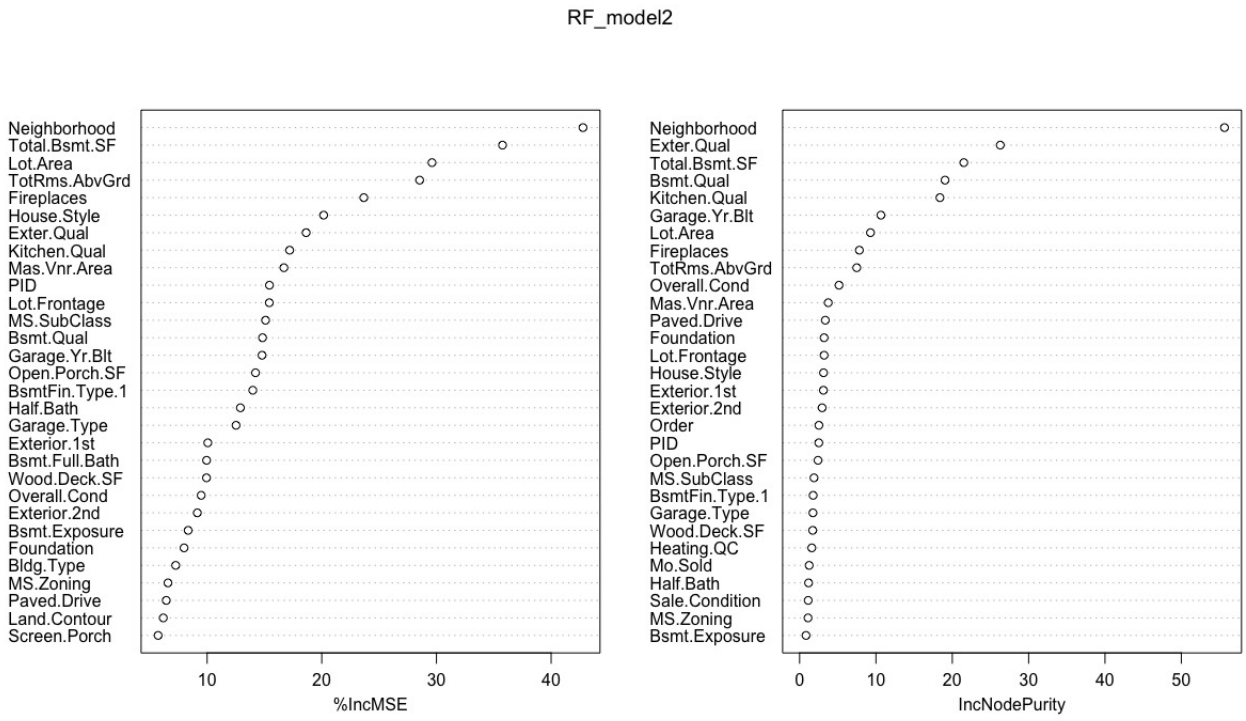

Figure 8: Random Forest on the Processed Data

From here I then ran multiple linear regression on the top 12 most important variable from the random forest.

4. Multiple Linear Regression

```
data <- data[, c("SalePrice",
                 "TotRms.AbvGrd",
                 "Neighborhood",
                 "Fireplaces",
                 "MS.SubClass",
                 "Total.Bsmt.SF",
                 "Exter.Qual",
                 "Lot.Area",
                 "Kitchen.Qual",
                 "House.Style",
                 "Mas.Vnr.Area",
                 "PID",
                 "Lot.Frontage"
                 )]
model1 <- lm(SalePrice ~., data = data)
summary(model1)
```

The output this created was quite messy and difficult to interpret as the neighborhood variable has 28 categories.

For this reason, I grouped the Neighborhood variable by the residuals in the model using the following code used in a tutorial in Term 1:

```
zip_groups <- data %>% # our dplyr code for creating a 3-level grouping for
    zipcode
  mutate(resid = resid(model1)) %>%
  group_by(Neighborhood) %>%
  summarise(median = median(resid),
            n = n()) %>%
  arrange(desc(median)) %>%
  mutate(sum_n = cumsum(n),
         ZipGroup = ntile(sum_n, 3))

data <- data %>% # joining the zipcode groups to our original dataset
  left_join(select(zip_groups, Neighborhood, ZipGroup), by = "Neighborhood")
```

I then replaced the Neighborhood variable with the ZipGroup, and reran the regression.

```
Call:
lm(formula = SalePrice ~ ., data = data)

Residuals:
     Min       1Q   Median       3Q      Max
-2.54405 -0.09018  0.00857  0.10951  0.67071

Coefficients:
                          Estimate        Std. Error t value             Pr(>|t|)
(Intercept)       11.46891092023249  0.03957645533157 289.791 < 0.0000000000000002 ***
TotRms.AbvGrd      0.03570455286286  0.00326768904132  10.927 < 0.0000000000000002 ***
ZipGroup           0.06469445964678  0.00541263079103  11.952 < 0.0000000000000002 ***
Fireplaces         0.10674730024137  0.00634259491387  16.830 < 0.0000000000000002 ***
MS.SubClass       -0.00065457591449  0.00010957061106  -5.974       0.000000002596 ***
Total.Bsmt.SF      0.00026160290290  0.00001234663097  21.188 < 0.0000000000000002 ***
Exter.QualGd       0.15884617378735  0.01171254894334  13.562 < 0.0000000000000002 ***
Exter.QualFa      -0.30310047203535  0.03464273914855  -8.749 < 0.0000000000000002 ***
Exter.QualEx       0.24260509636516  0.02731505513048   8.882 < 0.0000000000000002 ***
Lot.Area           0.00000330842855  0.00000051746077   6.394       0.000000000188 ***
Kitchen.QualFa    -0.40859111681425  0.03169969633288 -12.889 < 0.0000000000000002 ***
Kitchen.QualGd    -0.11672330278547  0.01881595278181  -6.203       0.000000000631 ***
Kitchen.QualPo    -0.36086984624860  0.19852550030252  -1.818             0.069205 .
Kitchen.QualTA    -0.26032415460133  0.02057452771589 -12.653 < 0.0000000000000002 ***
House.Style1.5Unf -0.09130022787161  0.04701941042979  -1.942             0.052263 .
House.Style1Story -0.04088058371125  0.01388971690133  -2.943             0.003274 **
House.Style2.5Fin  0.13519028135094  0.07204577702940   1.876             0.060694 .
House.Style2.5Unf  0.08966753406225  0.04233528409835   2.118             0.034257 *
House.Style2Story  0.09341866289868  0.01439840484588   6.488       0.000000000102 ***
House.StyleSFoyer  0.09629439979117  0.02539399522751   3.792             0.000152 ***
House.StyleSLvl    0.11645061570917  0.02142059471535   5.436       0.000000058879 ***
Mas.Vnr.Area       0.00009623511123  0.00002411772492   3.990       0.000067641897 ***
PID               -0.00000000009098  0.00000000002103  -4.326       0.000015718891 ***
Lot.Frontage       0.00046065923465  0.00021430784633   2.150             0.031675 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1971 on 2906 degrees of freedom
Multiple R-squared:  0.768,     Adjusted R-squared:  0.7662
F-statistic: 418.2 on 23 and 2906 DF,  p-value: < 0.00000000000000022
```

Figure 9: Regression, with Zipped Groups

I then used the avPlots() function from the "car" package in R to view some of the variables in the regression.

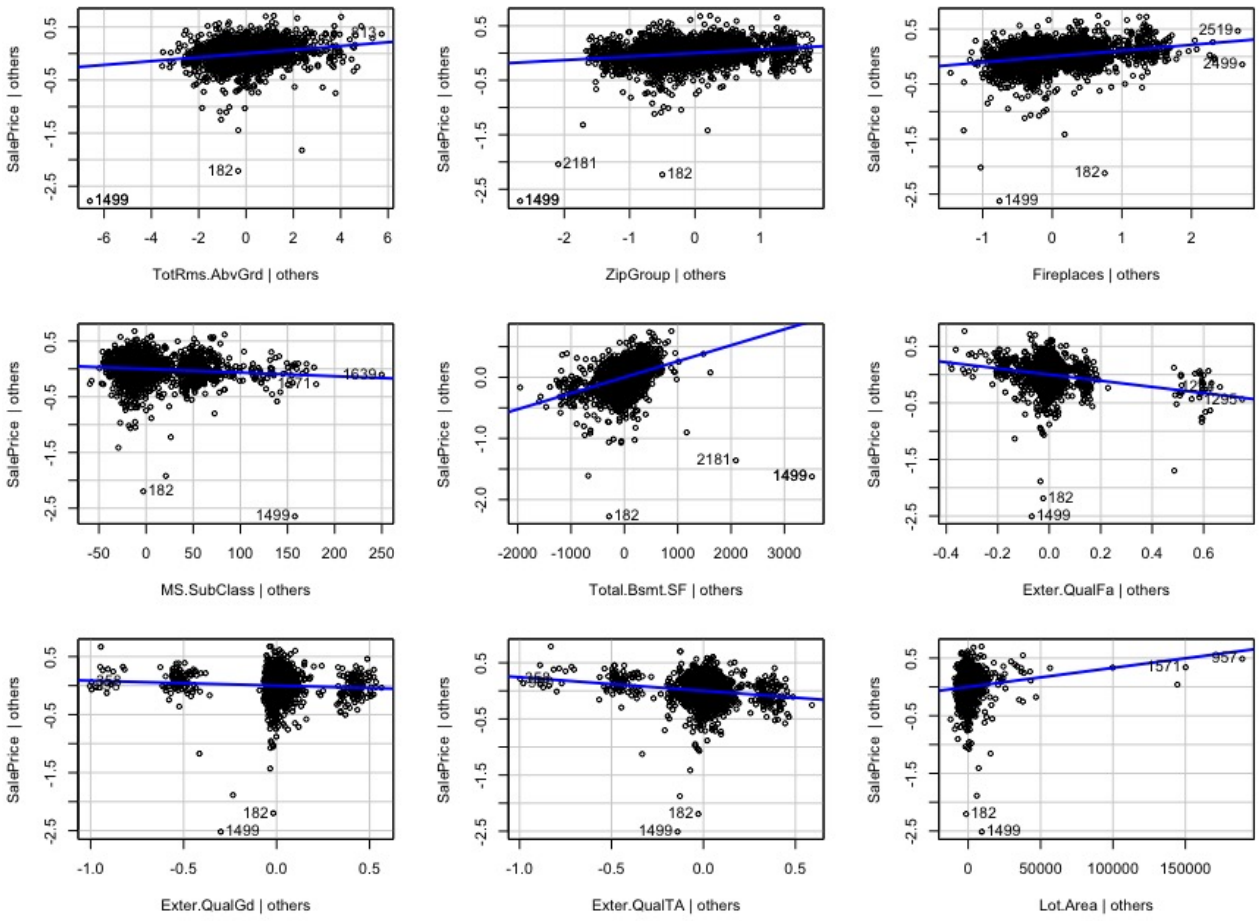

Figure 10: Regression Plot for some of the variables

4. Conclusions

The multivariate linear regression above explains less of the data than was described in the study, however this follows the correct prepossessing steps and is therefore representing a more accurate model.