STAT 844 Final Project Report

Winter 2018

Presented by:

Rongrong Su

Table of Contents

1. Introduction 4

2. Structure of data 4

2.1 Loading packages 4

2.2 Data structure 4

3. Data preprocessing – missing data, factorizing data and label encoding 7

3.1 Missing data 7

3.1.1 General idea 7

3.1.2 Pool related variables – Pool Quality and Pool Area 7

3.1.3 MiscFeature: Miscellaneous Features not covered in other categories 8

3.1.4 Alley: Type of alley access to property 8

3.1.5 Fence: Fence quality 9

3.1.6 Fireplace Group - FireplaceQC: Fireplace Quality Fireplaces:Number of fireplaces check if data make sense AND Fireplace 9

3.1.7 LotFrontage: Linear feet of street connected to property 11

3.1.8 Variables related to Garage 12

3.1.9 Variables related to Basement 16

3.1.10 Variables related to Masonry: MasVnrType and MasVnrArea 21

3.1.11 Electrical: Electrical system 23

3.2 Label encoding and factorizing the remaining character variables 24

3.2.1 MSZoning: General zoning classification of the sale 24

3.2.2 Street: Type of road access to property 24

3.2.3 LotShape:General shape of property 25

3.2.4 LandContour: Flatness of the property 25

3.2.5 Utilities: Type of utilities available 26

3.2.6 LotConfig: Lot configuration 26

3.2.7 LandSlope: Slope of property 27

3.2.8 Neighborhood: Physical locations within Ames city limits 27

3.2.9 Conditionl: Proximity to various conditions 27

3.2.10 Condition2: Proximity to various conditions 28

3.2.11 BldgType: Type of dwelling 29

3.2.12 HouseStyle: Style of dwelling 29

3.2.13 RoofStyle: Type of roof 30

3.2.14 RoofMatl:Roof material 30

3.2.15 Exterior covering on house 31

3.2.16 Exterior covering on house 32

3.2.17 Exterior covering on house 32

3.2.18 Evaluates the present condition of the material on the exterior 32

3.2.19 Foundation: Type of foundation 33

3.2.20 Heating: Type of heating 33

3.2.21 HeatingQC: Heating quality and condition 34

3.2.22 CentralAir: Central air conditioning 34

3.2.23 KitchenQual: Kitchen quality 35

3.2.24 Functional: Home functionality 35

3.2.25 PavedDrive: Paved driveway 36

3.2.26 SaleType: Type of sale 36

3.2.27 SaleCondition: Condition of sale 37
3.3 Changing some numeric variables into factor 38
3.3.1 YrSold: Year of sold 38
3.3.2 MoSold: Month of sold 38
3.3.3 MSSubClass: Identifies the type of dwelling involved in the sale. 38

4. Visualization of variables 39
4.1 Correlation and regression between numeric variables and SalePrice 39
4.2 Relation between SalePrice and variables related to area variables 43
4.3 Relation between SalePrice and variables related to Garage variables 44
4.4 Relation between SalePrice and variables related to Basement variables 45
4.5 Neighborhoods variables 47
4.6 MSSubClass 47
4.7 Relation between SalePrice and variables related to Quality variables 48

5. Feature engineering 50
5.1 Total Area 50
5.2 Total Bathroom 50
5.3 Consolidating Porch variables 50
5.4 House Age 50
5.5 The house is remodeled or not 51

6. Preparing data for modeling 51
6.1 Avoiding outliers 51
6.2 Dropping high correlated variables 51
6.3 Skewness and normalizing of numeric variable 52
6.4 Skewness and normalizing of response variables 53

7. Modeling 54
7.1 Random Forest – with IncNodePurity and IncMSE 54
7.2 GAM 57
7.3 Natural Cubic Spline and cross validation 58
7.4 Categorical regression model 63
7.5 Lasso with cross validation 63
7.6 Gradient Boosting 64
7.7 XGboost with cross validation 65

8. Conlusion 66

1. Introduction

This project is based on a Kaggle competition: House Prices: Advanced Regression Techniques. Kaggle describes this competition as follows:

Ask a home buyer to describe their dream house, and they probably won't begin with the height of the basement ceiling or the proximity to an east-west railroad. But this playground competition's dataset proves that much more influences price negotiations than the number of bedrooms or a white-picket fence.

With 79 explanatory variables describing (almost) every aspect of residential homes in Ames, Iowa, this competition challenges you to predict the final price of each home.

In brief, this project presents two parts: feature engineering and building model to predict the sale price of a house.

2. Structure of data

2.1 Loading packages

```
library(knitr)
library(ggplot2)
library(plyr)
library(dplyr)
library(corrplot)
library(caret)
library(gridExtra)
library(scales)
library(Rmisc)
library(ggrepel)
library(randomForest)
library(psych)
library(xgboost)
library(CatEncoders)
library(data.table)
library(mgcv)
library(DAAG)
library(crs)
library(Ckmeans.1d.dp)
library(splines)
#library(MASS)
```

2.2 Data structure

```
table<-read.csv('/Users/xhkj/Downloads/house-prices-advanced-regression-techn
iques/train.csv', stringsAsFactors = F)
str(table)
```

```
## 'data.frame':    1460 obs. of  81 variables:
##  $ Id           : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ MSSubClass   : int  60 20 60 70 60 50 20 60 50 190 ...
##  $ MSZoning     : chr  "RL" "RL" "RL" "RL" ...
##  $ LotFrontage  : int  65 80 68 60 84 85 75 NA 51 50 ...
##  $ LotArea      : int  8450 9600 11250 9550 14260 14115 10084 10382 6120 7
420 ...
##  $ Street       : chr  "Pave" "Pave" "Pave" "Pave" ...
##  $ Alley        : chr  NA NA NA NA ...
##  $ LotShape     : chr  "Reg" "Reg" "IR1" "IR1" ...
##  $ LandContour  : chr  "Lvl" "Lvl" "Lvl" "Lvl" ...
##  $ Utilities    : chr  "AllPub" "AllPub" "AllPub" "AllPub" ...
##  $ LotConfig    : chr  "Inside" "FR2" "Inside" "Corner" ...
##  $ LandSlope    : chr  "Gtl" "Gtl" "Gtl" "Gtl" ...
##  $ Neighborhood : chr  "CollgCr" "Veenker" "CollgCr" "Crawfor" ...
##  $ Condition1   : chr  "Norm" "Feedr" "Norm" "Norm" ...
##  $ Condition2   : chr  "Norm" "Norm" "Norm" "Norm" ...
##  $ BldgType     : chr  "1Fam" "1Fam" "1Fam" "1Fam" ...
##  $ HouseStyle   : chr  "2Story" "1Story" "2Story" "2Story" ...
##  $ OverallQual  : int  7 6 7 7 8 5 8 7 7 5 ...
##  $ OverallCond  : int  5 8 5 5 5 5 5 6 5 6 ...
##  $ YearBuilt    : int  2003 1976 2001 1915 2000 1993 2004 1973 1931 1939
 ...
##  $ YearRemodAdd : int  2003 1976 2002 1970 2000 1995 2005 1973 1950 1950
 ...
##  $ RoofStyle    : chr  "Gable" "Gable" "Gable" "Gable" ...
##  $ RoofMatl     : chr  "CompShg" "CompShg" "CompShg" "CompShg" ...
##  $ Exterior1st  : chr  "VinylSd" "MetalSd" "VinylSd" "Wd Sdng" ...
##  $ Exterior2nd  : chr  "VinylSd" "MetalSd" "VinylSd" "Wd Shng" ...
##  $ MasVnrType   : chr  "BrkFace" "None" "BrkFace" "None" ...
##  $ MasVnrArea   : int  196 0 162 0 350 0 186 240 0 0 ...
##  $ ExterQual    : chr  "Gd" "TA" "Gd" "TA" ...
##  $ ExterCond    : chr  "TA" "TA" "TA" "TA" ...
##  $ Foundation   : chr  "PConc" "CBlock" "PConc" "BrkTil" ...
##  $ BsmtQual     : chr  "Gd" "Gd" "Gd" "TA" ...
##  $ BsmtCond     : chr  "TA" "TA" "TA" "Gd" ...
##  $ BsmtExposure : chr  "No" "Gd" "Mn" "No" ...
##  $ BsmtFinType1 : chr  "GLQ" "ALQ" "GLQ" "ALQ" ...
##  $ BsmtFinSF1   : int  706 978 486 216 655 732 1369 859 0 851 ...
##  $ BsmtFinType2 : chr  "Unf" "Unf" "Unf" "Unf" ...
##  $ BsmtFinSF2   : int  0 0 0 0 0 0 0 32 0 0 ...
##  $ BsmtUnfSF    : int  150 284 434 540 490 64 317 216 952 140 ...
##  $ TotalBsmtSF  : int  856 1262 920 756 1145 796 1686 1107 952 991 ...
##  $ Heating      : chr  "GasA" "GasA" "GasA" "GasA" ...
##  $ HeatingQC    : chr  "Ex" "Ex" "Ex" "Gd" ...
##  $ CentralAir   : chr  "Y" "Y" "Y" "Y" ...
```

```
##  $ Electrical   : chr  "SBrkr" "SBrkr" "SBrkr" "SBrkr" ...
##  $ X1stFlrSF    : int  856 1262 920 961 1145 796 1694 1107 1022 1077 ...
##  $ X2ndFlrSF    : int  854 0 866 756 1053 566 0 983 752 0 ...
##  $ LowQualFinSF : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ GrLivArea    : int  1710 1262 1786 1717 2198 1362 1694 2090 1774 1077
 ...
##  $ BsmtFullBath : int  1 0 1 1 1 1 1 1 0 1 ...
##  $ BsmtHalfBath : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ FullBath     : int  2 2 2 1 2 1 2 2 2 1 ...
##  $ HalfBath     : int  1 0 1 0 1 1 0 1 0 0 ...
##  $ BedroomAbvGr : int  3 3 3 3 4 1 3 3 2 2 ...
##  $ KitchenAbvGr : int  1 1 1 1 1 1 1 1 2 2 ...
##  $ KitchenQual  : chr  "Gd" "TA" "Gd" "Gd" ...
##  $ TotRmsAbvGrd : int  8 6 6 7 9 5 7 7 8 5 ...
##  $ Functional   : chr  "Typ" "Typ" "Typ" "Typ" ...
##  $ Fireplaces   : int  0 1 1 1 1 0 1 2 2 2 ...
##  $ FireplaceQu  : chr  NA "TA" "TA" "Gd" ...
##  $ GarageType   : chr  "Attchd" "Attchd" "Attchd" "Detchd" ...
##  $ GarageYrBlt  : int  2003 1976 2001 1998 2000 1993 2004 1973 1931 1939
 ...
##  $ GarageFinish : chr  "RFn" "RFn" "RFn" "Unf" ...
##  $ GarageCars   : int  2 2 2 3 3 2 2 2 2 1 ...
##  $ GarageArea   : int  548 460 608 642 836 480 636 484 468 205 ...
##  $ GarageQual   : chr  "TA" "TA" "TA" "TA" ...
##  $ GarageCond   : chr  "TA" "TA" "TA" "TA" ...
##  $ PavedDrive   : chr  "Y" "Y" "Y" "Y" ...
##  $ WoodDeckSF   : int  0 298 0 0 192 40 255 235 90 0 ...
##  $ OpenPorchSF  : int  61 0 42 35 84 30 57 204 0 4 ...
##  $ EnclosedPorch: int  0 0 0 272 0 0 0 228 205 0 ...
##  $ X3SsnPorch   : int  0 0 0 0 0 320 0 0 0 0 ...
##  $ ScreenPorch  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolArea     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolQC       : chr  NA NA NA NA ...
##  $ Fence        : chr  NA NA NA NA ...
##  $ MiscFeature  : chr  NA NA NA NA ...
##  $ MiscVal      : int  0 0 0 0 0 700 0 350 0 0 ...
##  $ MoSold       : int  2 5 9 2 12 10 8 11 4 1 ...
##  $ YrSold       : int  2008 2007 2008 2006 2008 2009 2007 2009 2008 2008
 ...
##  $ SaleType     : chr  "WD" "WD" "WD" "WD" ...
##  $ SaleCondition: chr  "Normal" "Normal" "Normal" "Abnorml" ...
##  $ SalePrice    : int  208500 181500 223500 140000 250000 143000 307000 20
0000 129900 118000 ...
```

The data set contains 1460 records and 81 variables. Among 81 variables, there are character and integer variables and the last variable (SalePrice) is the response variable. The code and results are in appendix.

I notice that the ID variables is useless so I delete this column. And then I count the number of numeric and character variables. As can be seen, there are 37 numeric variables (including SalePrice) and 43 categorical variables.

3. Data preprocessing – missing data, factorizing data and label encoding

3.1 Missing data

```
na.cols <- which(colSums(is.na(table)) > 0)
sort(colSums(sapply(table[na.cols], is.na)), decreasing = TRUE)
```

```
##       PoolQC  MiscFeature        Alley        Fence  FireplaceQu
##         1453         1406         1369         1179          690
##  LotFrontage   GarageType  GarageYrBlt GarageFinish   GarageQual
##          259           81           81           81           81
##   GarageCond BsmtExposure BsmtFinType2     BsmtQual     BsmtCond
##           81           38           38           37           37
## BsmtFinType1   MasVnrType   MasVnrArea   Electrical
##           37            8            8            1
```

3.1.1 General idea

First I take a look at which variables contain missing value. There are 19 variables with missing value. Then I will fix each of them. And since some of variables are related like GarageType, GarageYrBlt I will fix them as a group. In the following analysis, I have converted character variables into ordinal integers if there is clear ordinal relation. And for the numerical missing value, I try to impute them with mean or median based on their meaning.

3.1.2 Pool related variables – Pool Quality and Pool Area

```
table %>% group_by(PoolQC) %>% count()
```

```
## # A tibble: 4 x 2
## # Groups:   PoolQC [4]
##   PoolQC     n
##   <chr>  <int>
## 1 <NA>    1453
## 2 Ex         2
## 3 Fa         2
## 4 Gd         3
```

First I look at PoolQC. PoolQC have four values: excellent, good, fair, no pool. It can be concluded that the missing values of NA of PoolQC with 0 PoolArea means no pool. To prove this assumption, I check if there is record that has value of PooArea is greater than 0 and is not NA of PoolQC. It turns out there is no such record. So, I assign 'None' to NA.

```
a<-table %>% select(PoolQC,PoolArea) %>%
  filter(is.na(table$PoolQu) & table$PoolArea >0)
nrow(a)
```

```
## [1] 0
```

```
table$PoolQC[is.na(table$PoolQC)] <- 'None'
```

And I notice that there is ordinal relation of value of PoolQC, I revalue it. And at last I check the result after transformation.

```
level<-c('None'=0,'Po'=1,'Fa'=2,'TA'=3,'Gd'=4,'Ex'=5)
table$PoolQC<-as.integer(revalue(table$PoolQC,level))
```

```
## The following `from` values were not present in `x`: Po, TA
```

```
table %>% group_by(PoolQC) %>% count()
```

```
## # A tibble: 4 x 2
## # Groups:   PoolQC [4]
##   PoolQC     n
##    <int> <int>
## 1      0  1453
## 2      2     2
## 3      4     3
## 4      5     2
```

3.1.3 MiscFeature: Miscellaneous Features not covered in other categories

```
table %>% group_by(MiscFeature) %>% count()
```

```
## # A tibble: 5 x 2
## # Groups:   MiscFeature [5]
##   MiscFeature     n
##   <chr>       <int>
## 1 <NA>         1406
## 2 Gar2            2
## 3 Othr            2
## 4 Shed           49
## 5 TenC            1
```

As the values are not ordinal, I convert MiscFeature into a factor. And the meaning of values is given by as follows. And at last I check the result after transformation.
Na-None, Car2-2nd Garge, Othr-other, Shed-Shed:over 100 SF, TenC-Tennis Court

```
table$MiscFeature[is.na(table$MiscFeature)]<-'None'
table$MiscFeature <- as.factor(table$MiscFeature)
```

3.1.4 Alley: Type of alley access to property

```
table %>% group_by(Alley) %>% count()
```

```
## # A tibble: 3 x 2
## # Groups:   Alley [3]
##   Alley     n
##   <chr> <int>
```

```
## 1 <NA>   1369
## 2 Grvl     50
## 3 Pave     41
```

As the values are not ordinal, I convert Alley into a factor. And the meaning of values is given by as follows. And at last I check the result after transformation.
NA-No alley access, Grvl-Gravel, Pave-Paved

```
table$Alley[is.na(table$Alley)]<-'None'
table$Alley <- as.factor(table$Alley)
```

3.1.5 Fence: Fence quality

```
table %>% group_by(Fence) %>% count()
```

```
## # A tibble: 5 x 2
## # Groups:   Fence [5]
##   Fence     n
##   <chr> <int>
## 1 <NA>   1179
## 2 GdPrv    59
## 3 GdWo     54
## 4 MnPrv   157
## 5 MnWw     11
```

As the values are not ordinal, I convert Fence into a factor. And the meaning of values is given by as follows.
NA-No Fence, GdPrv-Good Privacy, GdWo-Good Wood, MnPrv-Minimum Privacy, MnWw-Minimum Wood/Wire
And at last I check the result after transformation.

```
table$Fence[is.na(table$Fence)] <- 'None'
table$Fence <- as.factor(table$Fence)
table %>% group_by(Fence) %>% count()
```

```
## # A tibble: 5 x 2
## # Groups:   Fence [5]
##   Fence     n
##   <fct> <int>
## 1 GdPrv    59
## 2 GdWo     54
## 3 MnPrv   157
## 4 MnWw     11
## 5 None   1179
```

3.1.6 Fireplace Group - FireplaceQC: Fireplace Quality Fireplaces:Number of fireplaces check if data make sense AND Fireplace

First I check if there is unreasonable record whose FireplaceQC is NA but Fireplace is not equal to 0.

```
a<-table %>% select(FireplaceQu,Fireplaces) %>%
  filter(is.na(table$FireplaceQu) & table$Fireplaces>0)
nrow(a)

## [1] 0
```

So, there is no such record. Next I check values of FireplaceQC.

```
table %>% group_by(FireplaceQu) %>% count()

## # A tibble: 6 x 2
## # Groups:   FireplaceQu [6]
##   FireplaceQu     n
##   <chr>       <int>
## 1 <NA>          690
## 2 Ex             24
## 3 Fa             33
## 4 Gd            380
## 5 Po             20
## 6 TA            313
```

The meaning of values is given by as follows.
NA-No Fireplace, Ex-Excellent, Fa-Fair, Gd-Good, Po-Poor, TA-Average
There is a trend that the house of better Fireplace quality the price of it would be higher. (There is one exception: the average price of None class is larger than that of Poor class, which does not meet my expectation)

```
table[!is.na(table$SalePrice),] %>%
  group_by(FireplaceQu) %>%
  summarise(average=mean(SalePrice),counts=n())

## # A tibble: 6 x 3
##   FireplaceQu average counts
##   <chr>         <dbl>  <int>
## 1 Ex          337712.     24
## 2 Fa          167298.     33
## 3 Gd          226351.    380
## 4 None        141331.    690
## 5 Po          129764.     20
## 6 TA          205723.    313
```

I assign 'None' to missing value. Since it is ordinal I revalue it as I did for PoolQC.

```
table$FireplaceQu[is.na(table$FireplaceQu)]<-'None'

table$FireplaceQu<-as.integer(revalue(table$FireplaceQu,level))
table %>% group_by(table$FireplaceQu) %>% count()

## # A tibble: 6 x 2
## # Groups:   table$FireplaceQu [6]
##   `table$FireplaceQu`     n
```

```
##                  <int> <int>
## 1                    0   690
## 2                    1    20
## 3                    2    33
## 4                    3   313
## 5                    4   380
## 6                    5    24
```

3.1.7 LotFrontage: Linear feet of street connected to property

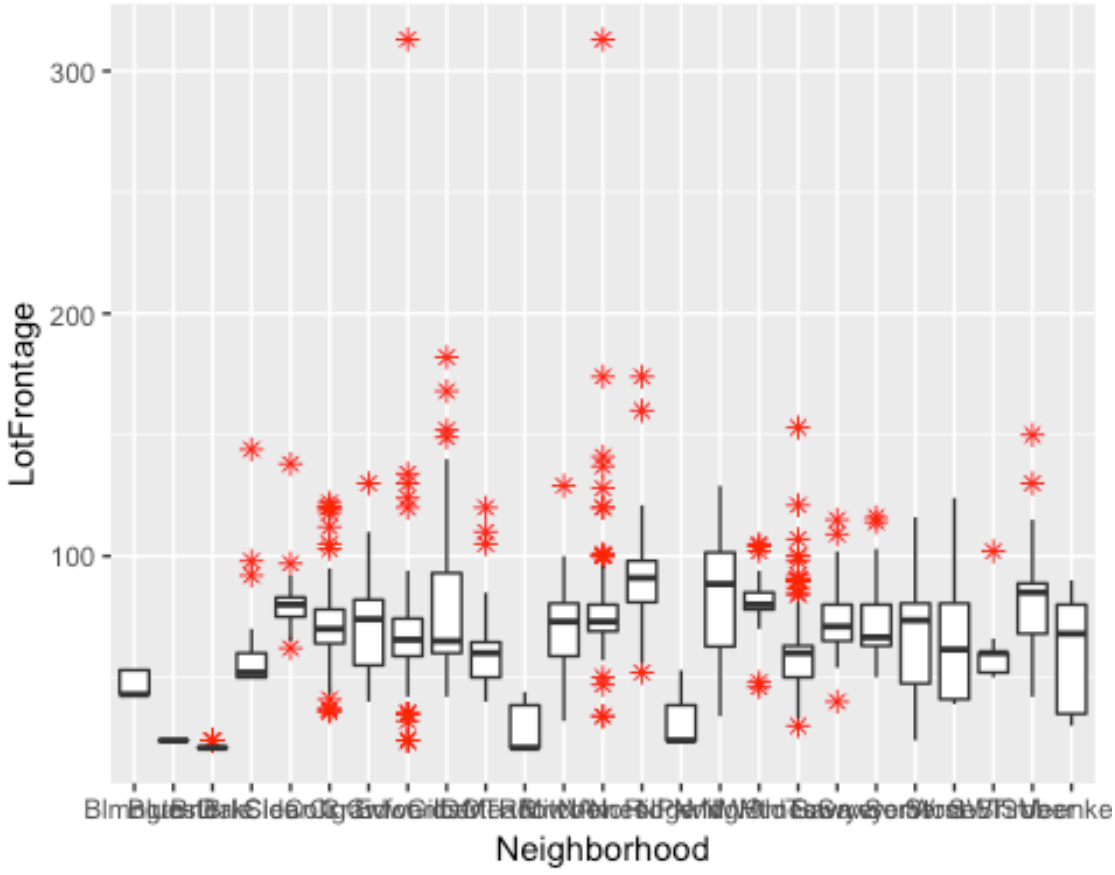
First it is a numerical variable.

```
typeof(table$LotFrontage)
```

```
## [1] "integer"
```

LotFrontage means the linear feet of street connected to property. It is unreasonable to simply assign 0 to missing value. However, normally, the houses in the same neighborhood have the similar distance to streets. So, I group by each neighborhood.

```
ggplot(table[!is.na(table$LotFrontage),], aes(x=Neighborhood,y=LotFrontage))+
  geom_boxplot(outlier.colour='red', outlier.shape=8,outlier.size=2)
```



Since there so many outliers, I decide to fill in the missing values by median LotFrontage of the neighborhood the house belongs to. And I check the result at last.

```
table[!is.na(table$LotFrontage),] %>%
  group_by(Neighborhood) %>%
  summarise(median=median(LotFrontage,na.rm=TRUE))
```

```
## # A tibble: 25 x 2
##    Neighborhood median
##    <chr>         <dbl>
##  1 Blmngtn        43
##  2 Blueste        24
##  3 BrDale         21
##  4 BrkSide        52
##  5 ClearCr        80
##  6 CollgCr        70
##  7 Crawfor        74
##  8 Edwards        65.5
##  9 Gilbert        65
## 10 IDOTRR         60
## # … with 15 more rows
```

```
table<-table %>%
  group_by(Neighborhood) %>%
  mutate(LotFrontage=replace(LotFrontage,is.na(LotFrontage),as.integer(median
(LotFrontage,na.rm =TRUE))))

sum(is.na(table$LotFrontage))
```

```
## [1] 0
```

3.1.8 Variables related to Garage

Variables related to Garage have missing value:
GarageType:Garage location
GarageYrBlt:Year garage was built
GarageFinish: Interior finish of the garage
GarageQual:Garage quality
GarageCond:Garage condition
Variables related to Garage do not have missing value:
GarageCars:Size of garage in car capacity
GarageArea: Size of garage in square feet
First I check if the missing values here mean houses does not have garages.

```
sum(is.na(table$GarageType))
```

```
## [1] 81
```

```
sum(is.na(table$GarageYrBlt))
```

```
## [1] 81
```

```
sum(is.na(table$GarageFinish))
```

```
## [1] 81
```

```
sum(is.na(table$GarageQual))
```

```
## [1] 81
```

```
sum(is.na(table$GarageCond))
```

```
## [1] 81
```

```
sum(table$GarageArea==0)
```

```
## [1] 81
```

```
sum(is.na(table$GarageCond) &
      is.na(table$GarageQual) &
      is.na(table$GarageFinish) &
      is.na(table$GarageYrBlt) &
      is.na(table$GarageType) &
      table$GarageArea==0 &
      table$GarageCars==0)
```

```
## [1] 81
```

As can be seen, the results are consistent with each other. So, I conclude that the missing values here mean the house does not have a garage.

3.1.8.1 GarageType: Garage location

```
table %>%
  group_by(GarageType) %>%
  count()
```

```
## # A tibble: 7 x 2
## # Groups:   GarageType [7]
##   GarageType     n
##   <chr>      <int>
## 1 <NA>          81
## 2 2Types         6
## 3 Attchd       870
## 4 Basment       19
## 5 BuiltIn       88
## 6 CarPort        9
## 7 Detchd       387
```

The meaning of values is given by the following.
NA-No Garage
2Types-More than one type of garage
Attchd-Attached to home
Basment-Basement Garage
BuiltIn-Built-In

CarPort-Car Port
Detchd-Detached from home
So, I fill NA with 'None' and factorize it.

```
table$GarageType[is.na(table$GarageType)] <- 'None'
table$GarageType <- as.factor(table$GarageType)
```

3.1.8.2 GarageYrBlt- Year garage was built

It is a numerical variable. So, I replace NA with the year house was built.

```
table$GarageYrBlt<-ifelse(is.na(table$GarageYrBlt),table$YearBuilt,table$Gara
geYrBlt)
sum(is.na(table$GarageYrBlt))

## [1] 0
```

3.1.8.3 GarageFinish- Interior finish of the garage

```
table %>%
  group_by(GarageFinish) %>%
  count()

## # A tibble: 4 x 2
## # Groups:   GarageFinish [4]
##   GarageFinish     n
##   <chr>        <int>
## 1 <NA>            81
## 2 Fin            352
## 3 RFn            422
## 4 Unf            605
```

And the meaning of values is given by the following.
NA-No Garage
Fin-Finished
RFn-Rough Finished
Unf-Unfinished
It is obvious that there is ordinal relation. So, I first assign 'None' to NA and revalue it.

```
table$GarageFinish[is.na(table$GarageFinish)] <- 'None'
level2<-c('None'=0, 'Unf'=1, 'RFn'=2, 'Fin'=3)
table$GarageFinish<-as.integer(revalue(table$GarageFinish,level2))

table %>%
  group_by(GarageFinish) %>%
  count()

## # A tibble: 4 x 2
## # Groups:   GarageFinish [4]
##   GarageFinish     n
```

```
##          <int> <int>
## 1            0    81
## 2            1   605
## 3            2   422
## 4            3   352
```

3.1.8.4 GarageQual:Garage quality

```
table %>%
  group_by(GarageQual) %>%
  count()

## # A tibble: 6 x 2
## # Groups:   GarageQual [6]
##   GarageQual     n
##   <chr>      <int>
## 1 <NA>          81
## 2 Ex             3
## 3 Fa            48
## 4 Gd            14
## 5 Po             3
## 6 TA          1311
```

I fill NA with 'None' and then revalue it.

```
table$GarageQual[is.na(table$GarageQual)]<-'None'
table$GarageQual<-as.integer(revalue(table$GarageQual,level))

table %>%
  group_by(GarageQual) %>%
  count()

## # A tibble: 6 x 2
## # Groups:   GarageQual [6]
##   GarageQual     n
##        <int> <int>
## 1          0    81
## 2          1     3
## 3          2    48
## 4          3  1311
## 5          4    14
## 6          5     3
```

3.1.8.5 GarageCond: Garage Condition

```
table %>%
  group_by(GarageCond) %>%
  count()

## # A tibble: 6 x 2
## # Groups:   GarageCond [6]
```

```
##   GarageCond     n
##   <chr>      <int>
## 1 <NA>          81
## 2 Ex             2
## 3 Fa            35
## 4 Gd             9
## 5 Po             7
## 6 TA          1326
```

```
table$GarageCond[is.na(table$GarageCond)] <- 'None'
table$GarageCond<-as.integer(revalue(table$GarageCond,level))
```

3.1.9 Variables related to Basement

Variables related to Garage have missing value:
BsmtExposure:Walkout or garden level walls BsmtFinType2:Quality of second finished area (if present)
BsmtQual:Height of the basement
BsmtCond:General condition of the basement BsmtFinType1:Quality of basement finished area
Variables related to Garage do not have missing value:
BsmtFinSF1: Type 1 finished square feet BsmtFinSF2: Type 2 finished square feet BsmtUnfSF: Unfinished square feet of basement area TotalBsmtSF: Total square feet of basement area
BsmtFullBath: Basement full bathrooms BsmtHalfBath: Basement half bathrooms
First I check if the missing values here mean houses does not have basements.

```
sum(is.na(table$BsmtExposure))
```

```
## [1] 38
```

```
sum(is.na(table$BsmtFinType2))
```

```
## [1] 38
```

```
sum(is.na(table$BsmtQual))
```

```
## [1] 37
```

```
sum(is.na(table$BsmtCond))
```

```
## [1] 37
```

```
sum(is.na(table$BsmtFinType1))
```

```
## [1] 37
```

```
sum(is.na(table$BsmtExposure) &
      is.na(table$BsmtFinType2) &
      is.na(table$BsmtQual) &
      is.na(table$BsmtCond) &
      is.na(table$BsmtFinType1))
```

```
## [1] 37
```

```
table<-data.frame(table)
table[!is.na(table$BsmtCond) & (is.na(table$BsmtFinType1)|is.na(table$BsmtQua
l)|is.na(table$BsmtExposure)|is.na(table$BsmtFinType2)), c('BsmtQual', 'BsmtC
ond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinType2')]

##     BsmtQual BsmtCond BsmtExposure BsmtFinType1 BsmtFinType2
## 333       Gd       TA           No          GLQ         <NA>
## 949       Gd       TA         <NA>          Unf          Unf
```

As can be seen, there are two house (333 and 949) with missing values which is not because of no basement. I impute the most common values for these two missing values.

```
table$BsmtFinType2[333] <- names(sort(-table(table$BsmtFinType2)))[1]
table$BsmtExposure[949] <- names(sort(-table(table$BsmtExposure)))[1]
```

Then I fix other ordinary variables.

3.1.9.1 BsmtQual: Height of the basement

A variable than can be made ordinal with the Qualities vector.

```
table %>%
  group_by(BsmtQual) %>%
  count()

## # A tibble: 5 x 2
## # Groups:   BsmtQual [5]
##   BsmtQual     n
##   <chr>    <int>
## 1 <NA>        37
## 2 Ex         121
## 3 Fa          35
## 4 Gd         618
## 5 TA         649

table$BsmtQual[is.na(table$BsmtQual)] <- 'None'
table$BsmtQual<-as.integer(revalue(table$BsmtQual, level))

## The following `from` values were not present in `x`: Po

table %>%
  group_by(BsmtQual) %>%
  count()

## # A tibble: 5 x 2
## # Groups:   BsmtQual [5]
##   BsmtQual     n
##      <int> <int>
## 1        0    37
## 2        2    35
## 3        3   649
```

```
## 4        4   618
## 5        5   121
```

3.1.9.2 BsmtCond: General condition of the basement

A variable than can be made ordinal with the Qualities vector.

```
table %>%
  group_by(BsmtCond) %>%
  count()

## # A tibble: 5 x 2
## # Groups:   BsmtCond [5]
##   BsmtCond     n
##   <chr>    <int>
## 1 <NA>        37
## 2 Fa          45
## 3 Gd          65
## 4 Po           2
## 5 TA        1311

table$BsmtCond[is.na(table$BsmtCond)] <- 'None'
table$BsmtCond<-as.integer(revalue(table$BsmtCond, level))

## The following `from` values were not present in `x`: Ex

table %>%
  group_by(BsmtCond) %>%
  count()

## # A tibble: 5 x 2
## # Groups:   BsmtCond [5]
##   BsmtCond     n
##      <int> <int>
## 1        0    37
## 2        1     2
## 3        2    45
## 4        3  1311
## 5        4    65
```

3.1.9.3 BsmtExposure: Walkout or garden level walls

A variable than can be made ordinal with the Qualities vector.

```
table %>%
  group_by(BsmtExposure) %>%
  count()

## # A tibble: 5 x 2
## # Groups:   BsmtExposure [5]
##   BsmtExposure     n
##   <chr>        <int>
```

```
## 1 <NA>            37
## 2 Av             221
## 3 Gd             134
## 4 Mn             114
## 5 No             954
```

NA-No Basement
No-No Exposure
Mn-Mimimum Exposure
Av-Average Exposure
Gd-Good Exposure

```
table$BsmtExposure[is.na(table$BsmtExposure)] <- 'None'
level3<-c('None'=0, 'No'=1, 'Mn'=2, 'Av'=3, 'Gd'=4)
table$BsmtExposure<-as.integer(revalue(table$BsmtExposure, level3))
table %>%
  group_by(BsmtExposure) %>%
  count()
```

```
## # A tibble: 5 x 2
## # Groups:   BsmtExposure [5]
##   BsmtExposure     n
##          <int> <int>
## 1            0    37
## 2            1   954
## 3            2   114
## 4            3   221
## 5            4   134
```

3.1.9.4 BsmtFinType1: Rating of basement finished area

A variable than can be made ordinal with the Qualities vector.

```
table %>%
  group_by(BsmtFinType1) %>%
  count()
```

```
## # A tibble: 7 x 2
## # Groups:   BsmtFinType1 [7]
##   BsmtFinType1     n
##   <chr>        <int>
## 1 <NA>            37
## 2 ALQ            220
## 3 BLQ            148
## 4 GLQ            418
## 5 LwQ             74
## 6 Rec            133
## 7 Unf            430
```

GLQ - Good Living Quarters

ALQ - Average Living Quarters
BLQ - Below Average Living Quarters
Rec - Average Rec Room
LwQ - Low Quality
Unf - Unfinshed
NA - No Basement

```
table$BsmtFinType1[is.na(table$BsmtFinType1)] <- 'None'
level4 <- c('None'=0, 'Unf'=1, 'LwQ'=2, 'Rec'=3, 'BLQ'=4, 'ALQ'=5, 'GLQ'=6)
table$BsmtFinType1<-as.integer(revalue(table$BsmtFinType1, level4))
table %>%
  group_by(BsmtFinType1) %>%
  count()
```

```
## # A tibble: 7 x 2
## # Groups:   BsmtFinType1 [7]
##   BsmtFinType1     n
##          <int> <int>
## 1            0    37
## 2            1   430
## 3            2    74
## 4            3   133
## 5            4   148
## 6            5   220
## 7            6   418
```

3.1.9.5 BsmtFinType2: Rating of basement finished area

A variable than can be made ordinal with the Qualities vector.

```
table %>%
  group_by(BsmtFinType2) %>%
  count()
```

```
## # A tibble: 7 x 2
## # Groups:   BsmtFinType2 [7]
##   BsmtFinType2     n
##   <chr>        <int>
## 1 <NA>            37
## 2 ALQ             19
## 3 BLQ             33
## 4 GLQ             14
## 5 LwQ             46
## 6 Rec             54
## 7 Unf           1257
```

```
table$BsmtFinType2[is.na(table$BsmtFinType2)] <- 'None'
table$BsmtFinType2<-as.integer(revalue(table$BsmtFinType2, level4))
table %>%
```

```
  group_by(BsmtFinType2) %>%
  count()
```

```
## # A tibble: 7 x 2
## # Groups:   BsmtFinType2 [7]
##   BsmtFinType2     n
##          <int> <int>
## 1            0    37
## 2            1  1257
## 3            2    46
## 4            3    54
## 5            4    33
## 6            5    19
## 7            6    14
```

Finally, check the variables again after all transformations are done.

```
sum(is.na(table$BsmtExposure))
```

```
## [1] 0
```

```
sum(is.na(table$BsmtFinType2))
```

```
## [1] 0
```

```
sum(is.na(table$BsmtQual))
```

```
## [1] 0
```

```
sum(is.na(table$BsmtCond))
```

```
## [1] 0
```

```
sum(is.na(table$BsmtFinType1))
```

```
## [1] 0
```

3.1.10 Variables related to Masonry: MasVnrType and MasVnrArea

MasVnrType: Masonry veneer type
MasVnrArea: Masonry veneer area
Firstly, I check if the missing values mean houses do not have Masonry veneer.

```
sum(is.na(table$MasVnrType))
```

```
## [1] 8
```

```
sum(is.na(table$MasVnrArea))
```

```
## [1] 8
```

```
sum(is.na(table$MasVnrArea) & is.na(table$MasVnrType))
```

```
## [1] 8
```

3.1.10.1 MasVnrType: Masonry veneer type

```
table %>%
  group_by(MasVnrType) %>%
  count()

## # A tibble: 5 x 2
## # Groups:   MasVnrType [5]
##   MasVnrType     n
##   <chr>      <int>
## 1 <NA>           8
## 2 BrkCmn        15
## 3 BrkFace      445
## 4 None         864
## 5 Stone        128
```

NA - None
BrkCmn - Brick Common
BrkFace - Brick Face
CBlock - Cinder Block
None - None Stone Stone
Next I check if there is a trend of sale price for different level of type of masonry venner.

```
table$MasVnrType[is.na(table$MasVnrType)] <- 'None'

table[!is.na(table$SalePrice),] %>%
  group_by(MasVnrType) %>%
  summarise(mean=mean(SalePrice)) %>%
  arrange(mean)

## # A tibble: 4 x 2
##   MasVnrType    mean
##   <chr>        <dbl>
## 1 BrkCmn     146318.
## 2 None       156958.
## 3 BrkFace    204692.
## 4 Stone      265584.
```

Clearly, there is a trend. So, I revalue it.

```
level5 <- c('None'=0, 'BrkCmn'=1, 'BrkFace'=2, 'Stone'=3)
table$MasVnrType<-as.integer(revalue(table$MasVnrType, level5))
table %>%
  group_by(MasVnrType) %>%
  count()

## # A tibble: 4 x 2
## # Groups:   MasVnrType [4]
##   MasVnrType     n
##        <int> <int>
```

```
## 1          0   872
## 2          1    15
## 3          2   445
## 4          3   128
```

3.1.10.2 MasVnrArea:Masonry veneer area

```
table$MasVnrArea[is.na(table$MasVnrArea)] <-0
```

3.1.11 Electrical: Electrical system

NA values are categorical

```
table %>%
  group_by(Electrical) %>%
  count()
```

```
## # A tibble: 6 x 2
## # Groups:   Electrical [6]
##   Electrical     n
##   <chr>      <int>
## 1 <NA>           1
## 2 FuseA         94
## 3 FuseF         27
## 4 FuseP          3
## 5 Mix            1
## 6 SBrkr       1334
```

I will impute the most common values for these two missing values. Or here I can just delete them.

```
table$Electrical[is.na(table$Electrical)] <- names(sort(-table(table$Electric
al)))[1]
table$Electrical <- as.factor(table$Electrical)

table %>%
  group_by(Electrical) %>%
  count()
```

```
## # A tibble: 5 x 2
## # Groups:   Electrical [5]
##   Electrical     n
##   <fct>      <int>
## 1 FuseA         94
## 2 FuseF         27
## 3 FuseP          3
## 4 Mix            1
## 5 SBrkr       1335
```

3.2 Label encoding and factorizing the remaining character variables

After fixing variables with missing values, I need to factorize the rest of character variables.

```
factor_va <- names(table[,sapply(table, is.character)])
```

3.2.1 MSZoning: General zoning classification of the sale

A categorical variable

```
table %>%
  group_by(MSZoning) %>%
  count()
```

```
## # A tibble: 5 x 2
## # Groups:   MSZoning [5]
##   MSZoning     n
##   <chr>    <int>
## 1 C (all)     10
## 2 FV          65
## 3 RH          16
## 4 RL        1151
## 5 RM         218
```

C - Commercial
FV - Floating Village Residential
RH - Residential High Density
RL - Residential Low Density
RM - Residential Medium Density

```
table$MSZoning <- as.factor(table$MSZoning)
```

3.2.2 Street: Type of road access to property

A categorical variable

```
table %>%
  group_by(Street) %>%
  count()
```

```
## # A tibble: 2 x 2
## # Groups:   Street [2]
##   Street     n
##   <chr>  <int>
## 1 Grvl       6
## 2 Pave    1454
```

```
table$Street <- as.factor(table$Street)
```

3.2.3 LotShape:General shape of property

A variable than can be made ordinal with the Qualities vector.

```
table %>%
  group_by(LotShape) %>%
  count()
```

```
## # A tibble: 4 x 2
## # Groups:   LotShape [4]
##   LotShape     n
##   <chr>    <int>
## 1 IR1        484
## 2 IR2         41
## 3 IR3         10
## 4 Reg        925
```

level6:
Reg - Regular
IR1 - Slightly irregular
IR2 - Moderately Irregular
IR3 – Irregular

```
level6<-c('IR3'=0, 'IR2'=1, 'IR1'=2, 'Reg'=3)
table$LotShape<-as.integer(revalue(table$LotShape,level6 ))
```

```
table %>%
  group_by(LotShape) %>%
  count()
```

```
## # A tibble: 4 x 2
## # Groups:   LotShape [4]
##   LotShape     n
##      <int> <int>
## 1        0    10
## 2        1    41
## 3        2   484
## 4        3   925
```

3.2.4 LandContour: Flatness of the property

A categorical variable

```
table %>%
  group_by(LandContour) %>%
  count()
```

```
## # A tibble: 4 x 2
## # Groups:   LandContour [4]
```

```
##   LandContour     n
##   <chr>       <int>
## 1 Bnk            63
## 2 HLS            50
## 3 Low            36
## 4 Lvl          1311
```

Bnk - Banked(Quick and significant rise from street grade to building)

HLS - Hillside(Significant slope from side to side)

Low - Depression

Lvl - Near Flat/Level

```
table$LandContour <- as.factor(table$LandContour)
```

3.2.5 Utilities: Type of utilities available

A categorical variable

```
table %>%
  group_by(Utilities) %>%
  count()

## # A tibble: 2 x 2
## # Groups:   Utilities [2]
##   Utilities     n
##   <chr>     <int>
## 1 AllPub     1459
## 2 NoSeWa        1
```

This variable does not important so I drop it.

```
table$Utilities<-NULL
```

3.2.6 LotConfig: Lot configuration

A categorical variable

```
table %>%
  group_by(LotConfig) %>%
  count()

## # A tibble: 5 x 2
## # Groups:   LotConfig [5]
##   LotConfig     n
##   <chr>     <int>
## 1 Corner      263
## 2 CulDSac      94
## 3 FR2          47
```

```
## 4 FR3           4
## 5 Inside     1052
```

Corner - Corner lot
CulDSac - Cul-de-sac
FR2 - Frontage on 2 sides of property
FR3 - Frontage on 3 sides of property Inside
Inside – Inside lot

```
table$LotConfig <- as.factor(table$LotConfig)
```

3.2.7 LandSlope: Slope of property

A variable than can be made ordinal with the Qualities vector.

```
table %>%
  group_by(LandSlope) %>%
  count()

## # A tibble: 3 x 2
## # Groups:   LandSlope [3]
##   LandSlope     n
##   <chr>     <int>
## 1 Gtl        1382
## 2 Mod          65
## 3 Sev          13
```

level7:
Gtl - Gentle slope
Mod - Moderate Slope
Sev Severe Slope

```
level7<-c('Sev'=0, 'Mod'=1, 'Gtl'=2)
table$LandSlope<-as.integer(revalue(table$LandSlope,level7))
```

3.2.8 Neighborhood: Physical locations within Ames city limits

A categorical variable

```
table$Neighborhood <- as.factor(table$Neighborhood)
```

3.2.9 Conditionl: Proximity to various conditions

A categorical variable

```
table %>%
  group_by(Condition1) %>%
  count()
```

```
## # A tibble: 9 x 2
## # Groups:   Condition1 [9]
##   Condition1     n
##   <chr>      <int>
## 1 Artery        48
## 2 Feedr         81
## 3 Norm        1260
## 4 PosA           8
## 5 PosN          19
## 6 RRAe          11
## 7 RRAn          26
## 8 RRNe           2
## 9 RRNn           5
```

Artery - Adjacent to arterial street
Feedr - Adjacent to feeder street
Norm - Normal
RRNn - Within 200' of North-South Railroad
RRAn - Adjacent to North-South Railroad
PosN - Near positive off-site feature–park, greenbelt, etc.
PosA - Adjacent to postive off-site feature
RRNe - Within 200' of East-West Railroad
RRAe - Adjacent to East-West Railroad

```
table$Condition1 <- as.factor(table$Condition1)
```

3.2.10 Condition2: Proximity to various conditions

A categorical variable

```
table %>%
  group_by(Condition2) %>%
  count()
```

```
## # A tibble: 8 x 2
## # Groups:   Condition2 [8]
##   Condition2     n
##   <chr>      <int>
## 1 Artery         2
## 2 Feedr          6
## 3 Norm        1445
## 4 PosA           1
## 5 PosN           2
## 6 RRAe           1
## 7 RRAn           1
## 8 RRNn           2
```

```
table$Condition2 <- as.factor(table$Condition2)
```

3.2.11 BldgType: Type of dwelling

A categorical variable

```
table %>%
  group_by(BldgType) %>%
  count()
```

```
## # A tibble: 5 x 2
## # Groups:   BldgType [5]
##   BldgType     n
##   <chr>    <int>
## 1 1Fam      1220
## 2 2fmCon      31
## 3 Duplex      52
## 4 Twnhs       43
## 5 TwnhsE     114
```

1Fam - Single-family Detached
2FmCon - Two-family Conversion; originally built as one-family dwelling
Duplx - Duplex
TwnhsE - Townhouse End Unit
TwnhsI - Townhouse Inside Unit

```
table$BldgType <- as.factor(table$BldgType)
```

3.2.12 HouseStyle: Style of dwelling

A categorical variable

```
table %>%
  group_by(HouseStyle) %>%
  count()
```

```
## # A tibble: 8 x 2
## # Groups:   HouseStyle [8]
##   HouseStyle     n
##   <chr>      <int>
## 1 1.5Fin       154
## 2 1.5Unf        14
## 3 1Story       726
## 4 2.5Fin         8
## 5 2.5Unf        11
## 6 2Story       445
## 7 SFoyer        37
## 8 SLvl          65
```

1Story - One story
1.5Fin - One and one-half story: 2nd level finished
1.5Unf - One and one-half story: 2nd level unfinished 2Story Two story

2.5Fin - Two and one-half story: 2nd level finished
2.5Unf - Two and one-half story: 2nd level unfinished
SFoyer - Split Foyer
SLvl - Split Level

```
table$HouseStyle <- as.factor(table$HouseStyle)
```

3.2.13 RoofStyle: Type of roof

A categorical variable

```
table %>%
  group_by(RoofStyle) %>%
  count()
```

```
## # A tibble: 6 x 2
## # Groups:   RoofStyle [6]
##   RoofStyle     n
##   <chr>     <int>
## 1 Flat         13
## 2 Gable      1141
## 3 Gambrel      11
## 4 Hip         286
## 5 Mansard       7
## 6 Shed          2
```

Flat - Flat
Gable - Gable
Gambrel - Gabrel (Barn)
Hip - Hip
Mansard - Mansard
Shed – Shed

```
table$RoofStyle <- as.factor(table$RoofStyle
```

3.2.14 RoofMatl:Roof material

A categorical variable

```
table %>%
  group_by(RoofMatl) %>%
  count()
```

```
## # A tibble: 8 x 2
## # Groups:   RoofMatl [8]
##   RoofMatl     n
##   <chr>    <int>
## 1 ClyTile      1
## 2 CompShg   1434
## 3 Membran      1
```

```
## 4 Metal        1
## 5 Roll         1
## 6 Tar&Grv     11
## 7 WdShake      5
## 8 WdShngl      6
```

ClyTile - Clay or Tile
CompShg - Standard (Composite) Shingle
Membran - Membrane
Metal - Metal
Roll – Roll
Tar&Grv - Gravel & Tar
WdShake - Wood Shakes
WdShngl - Wood Shingles

```
table$RoofMatl <- as.factor(table$RoofMatl)
```

3.2.15 Exterior covering on house

A categorical variable

```
table %>%
  group_by(Exterior1st) %>%
  count()
```

```
## # A tibble: 15 x 2
## # Groups:   Exterior1st [15]
##    Exterior1st     n
##    <chr>       <int>
##  1 AsbShng        20
##  2 AsphShn         1
##  3 BrkComm         2
##  4 BrkFace        50
##  5 CBlock          1
##  6 CemntBd        61
##  7 HdBoard       222
##  8 ImStucc         1
##  9 MetalSd       220
## 10 Plywood       108
## 11 Stone           2
## 12 Stucco         25
## 13 VinylSd       515
## 14 Wd Sdng       206
## 15 WdShing        26
```

AsbShng - Asbestos Shingles
AsphShn - Asphalt Shingles
BrkComm - Brick Common
BrkFace - Brick Face

CBlock - Cinder Block
CemntBd - Cement Board
HdBoard - Hard Board
ImStucc - Imitation Stucco
MetalSd - Metal Siding
Other - Other
Plywood - Plywood
PreCast - PreCast
Stone - Stone
Stucco - Stucco
VinylSd - Vinyl Siding
Wd Sdng - Wood Siding
WdShing - Wood Shingles

```
table$Exterior1st <- as.factor(table$Exterior1st)
```

3.2.16 Exterior covering on house

A variable than can be made ordinal with the Qualities vector.

```
table %>%
  group_by(ExterQual) %>%
  count()
```

```
## # A tibble: 4 x 2
## # Groups:   ExterQual [4]
##   ExterQual     n
##   <chr>     <int>
## 1 Ex           52
## 2 Fa           14
## 3 Gd          488
## 4 TA          906
```

```
table$ExterQual<-as.integer(revalue(table$ExterQual, level))
```

```
## The following `from` values were not present in `x`: None, Po
```

3.2.17 Exterior covering on house

A categorical variable

```
table$Exterior2nd <- as.factor(table$Exterior2nd)
```

3.2.18 Evaluates the present condition of the material on the exterior

A variable than can be made ordinal with the Qualities vector.

```
table %>%
  group_by(ExterCond) %>%
  count()
```

```
## # A tibble: 5 x 2
## # Groups:   ExterCond [5]
##   ExterCond     n
##   <chr>     <int>
## 1 Ex            3
## 2 Fa           28
## 3 Gd          146
## 4 Po            1
## 5 TA         1282
```

```
table$ExterCond<-as.integer(revalue(table$ExterCond, level))
```

```
## The following `from` values were not present in `x`: None
```

3.2.19 Foundation: Type of foundation

A categorical variable.

```
table %>%
  group_by(Foundation) %>%
  count()
```

```
## # A tibble: 6 x 2
## # Groups:   Foundation [6]
##   Foundation     n
##   <chr>      <int>
## 1 BrkTil       146
## 2 CBlock       634
## 3 PConc        647
## 4 Slab          24
## 5 Stone          6
## 6 Wood           3
```

BrkTil - Brick & Tile
CBlock - Cinder Block
PConc - Poured Contrete
Slab - Slab
Stone - Stone
Wood – Wood

```
table$Foundation <- as.factor(table$Foundation)
```

3.2.20 Heating: Type of heating

A categorical variable.

```
table %>%
  group_by(Heating) %>%
  count()
```

```
## # A tibble: 6 x 2
## # Groups:   Heating [6]
##   Heating     n
##   <chr>   <int>
## 1 Floor       1
## 2 GasA     1428
## 3 GasW       18
## 4 Grav        7
## 5 OthW        2
## 6 Wall        4
```

Floor - Floor Furnace
GasA - Gas forced warm air furnace
GasW - Gas hot water or steam heat
Grav - Gravity furnace
OthW - Hot water or steam heat other than gas
Wall - Wall furnace

```
table$Heating <- as.factor(table$Heating)
```

3.2.21 HeatingQC: Heating quality and condition

A variable than can be made ordinal with the Qualities vector.

```
table %>%
  group_by(HeatingQC) %>%
  count()
```

```
## # A tibble: 5 x 2
## # Groups:   HeatingQC [5]
##   HeatingQC     n
##   <chr>     <int>
## 1 Ex          741
## 2 Fa           49
## 3 Gd          241
## 4 Po            1
## 5 TA          428
```

```
table$HeatingQC<-as.integer(revalue(table$HeatingQC, level))
```

```
## The following `from` values were not present in `x`: None
```

3.2.22 CentralAir: Central air conditioning

A categorical variable. Since values of it only have two level, I will revalute it with 0 and 1.

```
table %>%
  group_by(CentralAir) %>%
  count()
```

```
## # A tibble: 2 x 2
## # Groups:   CentralAir [2]
##   CentralAir     n
##   <chr>      <int>
## 1 N             95
## 2 Y           1365
```

```
table$CentralAir<-as.integer(revalue(table$CentralAir, c('N'=0, 'Y'=1)))
```

3.2.23 KitchenQual: Kitchen quality

A variable than can be made ordinal with the Qualities vector.

```
table %>%
  group_by(KitchenQual) %>%
  count()
```

```
## # A tibble: 4 x 2
## # Groups:   KitchenQual [4]
##   KitchenQual     n
##   <chr>       <int>
## 1 Ex            100
## 2 Fa             39
## 3 Gd            586
## 4 TA            735
```

```
table$KitchenQual<-as.integer(revalue(table$KitchenQual, level))
```

```
## The following `from` values were not present in `x`: None, Po
```

3.2.24 Functional: Home functionality

A variable than can be made ordinal with the Qualities vector.

```
table %>%
  group_by(Functional) %>%
  count()
```

```
## # A tibble: 7 x 2
## # Groups:   Functional [7]
##   Functional     n
##   <chr>      <int>
## 1 Maj1          14
## 2 Maj2           5
## 3 Min1          31
```

```
## 4 Min2          34
## 5 Mod           15
## 6 Sev            1
## 7 Typ         1360
```

level 8:
Typ - Typical Functionality
Min1 - Minor Deductions 1
Min2 - Minor Deductions 2
Mod - Moderate Deductions
Maj1 - Major Deductions 1
Maj2 - Major Deductions 2
Sev - Severely Damaged Sal Salvage only

```
level8<-c('Sal'=0, 'Sev'=1, 'Maj2'=2, 'Maj1'=3, 'Mod'=4, 'Min2'=5, 'Min1'=6,
'Typ'=7)
table$Functional <- as.integer(revalue(table$Functional,level8))

## The following `from` values were not present in `x`: Sal
```

3.2.25 PavedDrive: Paved driveway

A variable than can be made ordinal with the Qualities vector.

```
table %>%
  group_by(PavedDrive) %>%
  count()

## # A tibble: 3 x 2
## # Groups:   PavedDrive [3]
##   PavedDrive     n
##   <chr>      <int>
## 1 N             90
## 2 P             30
## 3 Y           1340
```

level 9
Y - Paved
P - Partial Pavement
N - Dirt/Gravel

```
level9<-c('N'=0, 'P'=1, 'Y'=2)
table$PavedDrive<-as.integer(revalue(table$PavedDrive,level9))
```

3.2.26 SaleType: Type of sale

A categorical variable

```
table %>%
  group_by(SaleType) %>%
  count()
```

```
## # A tibble: 9 x 2
## # Groups:   SaleType [9]
##   SaleType     n
##   <chr>    <int>
## 1 COD         43
## 2 Con          2
## 3 ConLD        9
## 4 ConLI        5
## 5 ConLw        5
## 6 CWD          4
## 7 New        122
## 8 Oth          3
## 9 WD        1267
```

WD - Warranty Deed - Conventional
CWD - Warranty Deed - Cash
VWD - Warranty Deed - VA Loan
New - Home just constructed and sold
COD - Court Officer Deed/Estate Con Contract 15% Down payment regular terms
ConLw - Contract Low Down payment and low interest
ConLI - Contract Low Interest
ConLD - Contract Low Down
Oth – Other

```
table$SaleType <- as.factor(table$SaleType)
```

3.2.27 SaleCondition: Condition of sale

A categorical variable

```
table %>%
  group_by(SaleCondition) %>%
  count()
```

```
## # A tibble: 6 x 2
## # Groups:   SaleCondition [6]
##   SaleCondition     n
##   <chr>         <int>
## 1 Abnorml         101
## 2 AdjLand           4
## 3 Alloca           12
## 4 Family           20
## 5 Normal         1198
## 6 Partial         125
```

Normal - Normal Sale
Abnorml - Abnormal Sale (trade, foreclosure, short sale)
AdjLand - Adjoining Land Purchase
Alloca - Allocation
Family - Sale between family members
Partial - Home was not completed when last assessed

```
table$SaleCondition <- as.factor(table$SaleCondition)
```

3.3 Changing some numeric variables into factor

At this point, all the missing values and character variables have been processed. But there are there numeric variables need to be converted into factor: YrSold, MoSold and MSSubclass. Since it does not make sense to treat them as numeric variables.

3.3.1 YrSold: Year of sold

```
str(table$YrSold)
```

```
##  int [1:1460] 2008 2007 2008 2006 2008 2009 2007 2009 2008 2008 ...
```

```
table$MoSold <- as.factor(table$MoSold)
```

3.3.2 MoSold: Month of sold

```
str(table$MoSold)
```

```
##  Factor w/ 12 levels "1","2","3","4",..: 2 5 9 2 12 10 8 11 4 1 ...
```

```
table$MoSold <- as.factor(table$MoSold)
```

3.3.3 MSSubClass: Identifies the type of dwelling involved in the sale.

```
table %>%
  group_by(MSSubClass) %>%
  count()
```

```
## # A tibble: 15 x 2
## # Groups:   MSSubClass [15]
##    MSSubClass     n
##         <int> <int>
##  1         20   536
##  2         30    69
##  3         40     4
##  4         45    12
##  5         50   144
##  6         60   299
##  7         70    60
##  8         75    16
```

```
##  9          80    58
## 10          85    20
## 11          90    52
## 12         120    87
## 13         160    63
## 14         180    10
## 15         190    30
```

level 10:
20 1-STORY 1946 & NEWER ALL STYLES
30 1-STORY 1945 & OLDER
40 1-STORY W/FINISHED ATTIC ALL AGES
45 1-1/2 STORY - UNFINISHED ALL AGES
50 1-1/2 STORY FINISHED ALL AGES
60 2-STORY 1946 & NEWER
70 2-STORY 1945 & OLDER
75 2-1/2 STORY ALL AGES
80 SPLIT OR MULTI-LEVEL 85 SPLIT FOYER
90 DUPLEX - ALL STYLES AND AGES
120 1-STORY PUD (Planned Unit Development) - 1946 & NEWER
150 1-1/2 STORY PUD - ALL AGES
160 2-STORY PUD - 1946 & NEWER
180 PUD - MULTILEVEL - INCL SPLIT LEV/FOYER
190 2 FAMILY CONVERSION - ALL STYLES AND AGES

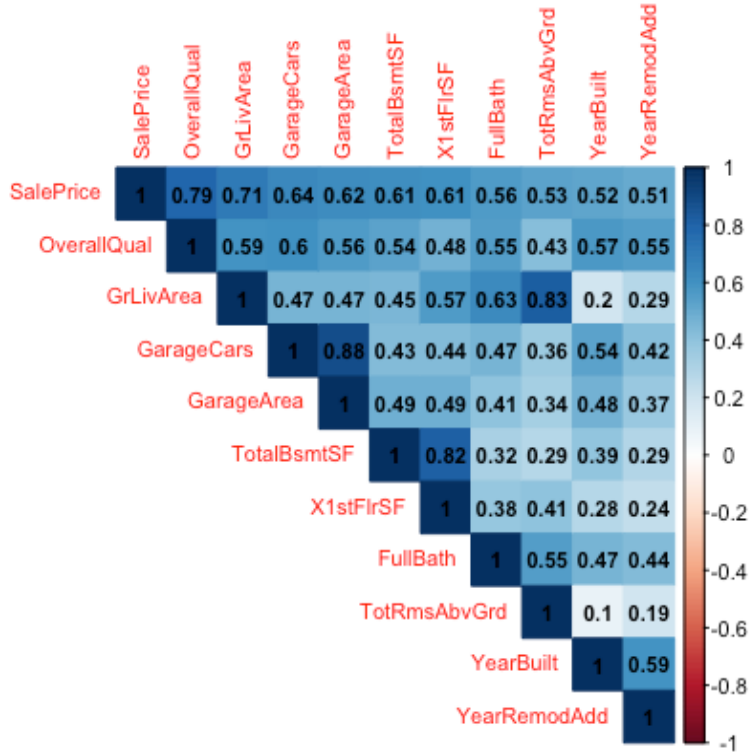
```
level10<-c('20'='1 story 1946+', '30'='1 story 1945-', '40'='1 story unf atti
c', '45'='1,5 story unf', '50'='1,5 story fin', '60'='2 story 1946+', '70'='2
 story 1945-', '75'='2,5 story all ages', '80'='split/multi level', '85'='spl
it foyer', '90'='duplex all style/age', '120'='1 story PUD 1946+', '150'='1,5
 story PUD all', '160'='2 story PUD 1946+', '180'='PUD multilevel', '190'='2
family conversion')
table$MSSubClass<-as.factor(table$MSSubClass)
table$MSSubClass <- revalue(table$MSSubClass,level10)

## The following `from` values were not present in `x`: 150

str(table$MSSubClass)

##  Factor w/ 15 levels "1 story 1946+",..: 6 1 6 7 6 5 1 6 5 15 ...
```

4. Visualization of variables

4.1 Correlation and regression between numeric variables and SalePrice

```
numvar<-table[,num_var]
cor_numvar<-cor(numvar)
sorted_cor <- as.matrix(sort(cor_numvar[,'SalePrice'], decreasing = TRUE))
effecive_var<-names(which(apply(sorted_cor, 1, function(x) abs(x)>0.5)))
corrplot(as.matrix(cor_numvar[effecive_var,effecive_var]),
         type = 'upper',
```

```
         method='color',
         addCoef.col = 'black',
         tl.cex = .7,
         cl.cex = .7,
         number.cex=.7)
```



As can be seen, the Overall Quality and Above grade living area have highest correlation with SalePrice. Clearly, it is a multicollinearity issue. For example, the correlation between Above grade living area and Total basement above grade living area is 0.83. And the correlation between Total basement above grade living area and SalePrice is 0.53. And there are other cases. So, I need to address this issue by eliminating some variables. And below is the distribution and regression between each variable and SalePrice.

```
require(GGally)
```

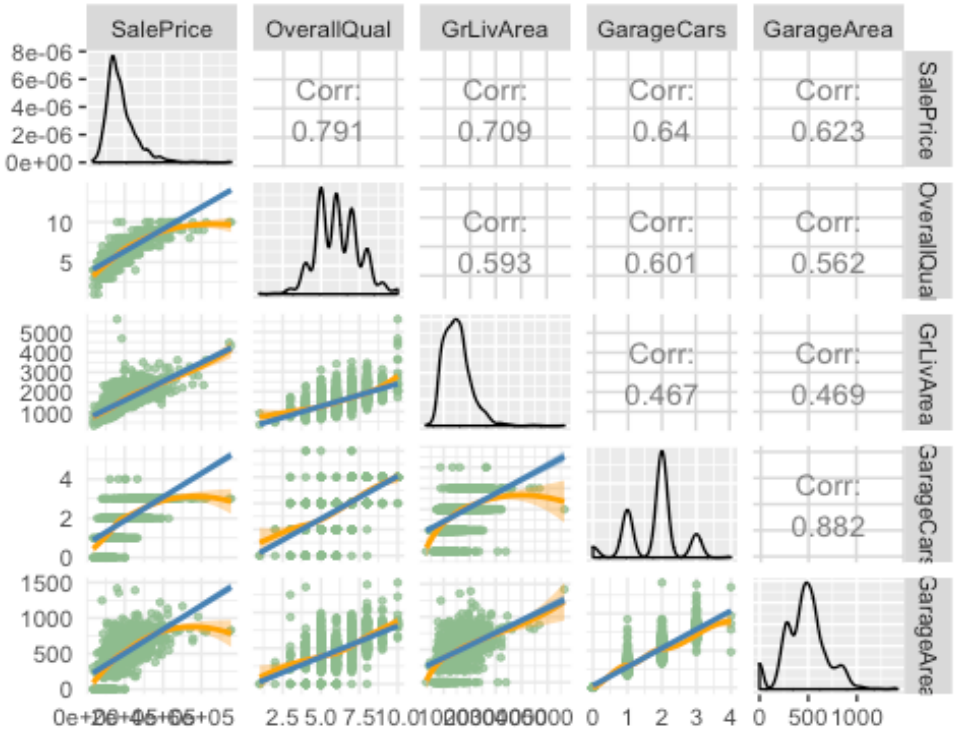
```
## Loading required package: GGally
```

```
##
## Attaching package: 'GGally'
```

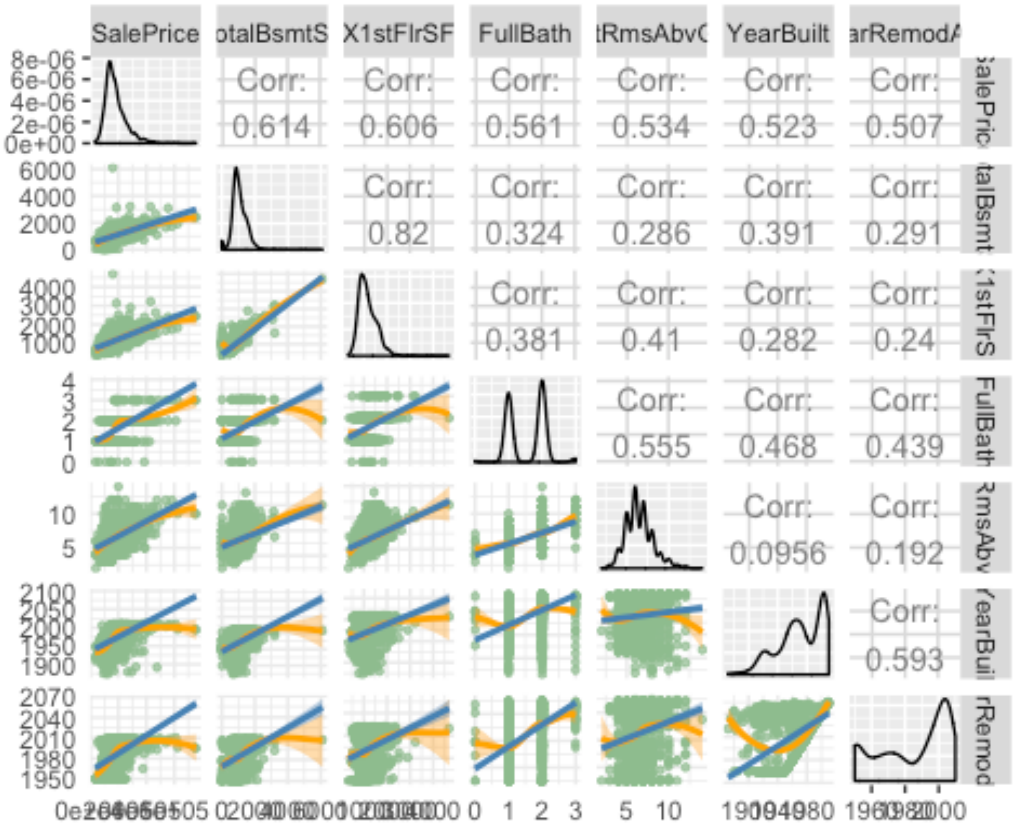
```
## The following object is masked from 'package:dplyr':
##
##     nasa
```

```
lm.plt <- function(data, mapping, ...){
   plt <- ggplot(data = data, mapping = mapping) +
    geom_point(shape = 20, alpha = 0.7, color = 'darkseagreen') +
    geom_smooth(method=loess, fill='orange', color='orange') +
    geom_smooth(method=lm, fill='steelblue', color='steelblue') +
```

```
    theme_minimal()
  return(plt)
}
ggpairs(table, effecive_var[1:5], lower = list(continuous = lm.plt))
```

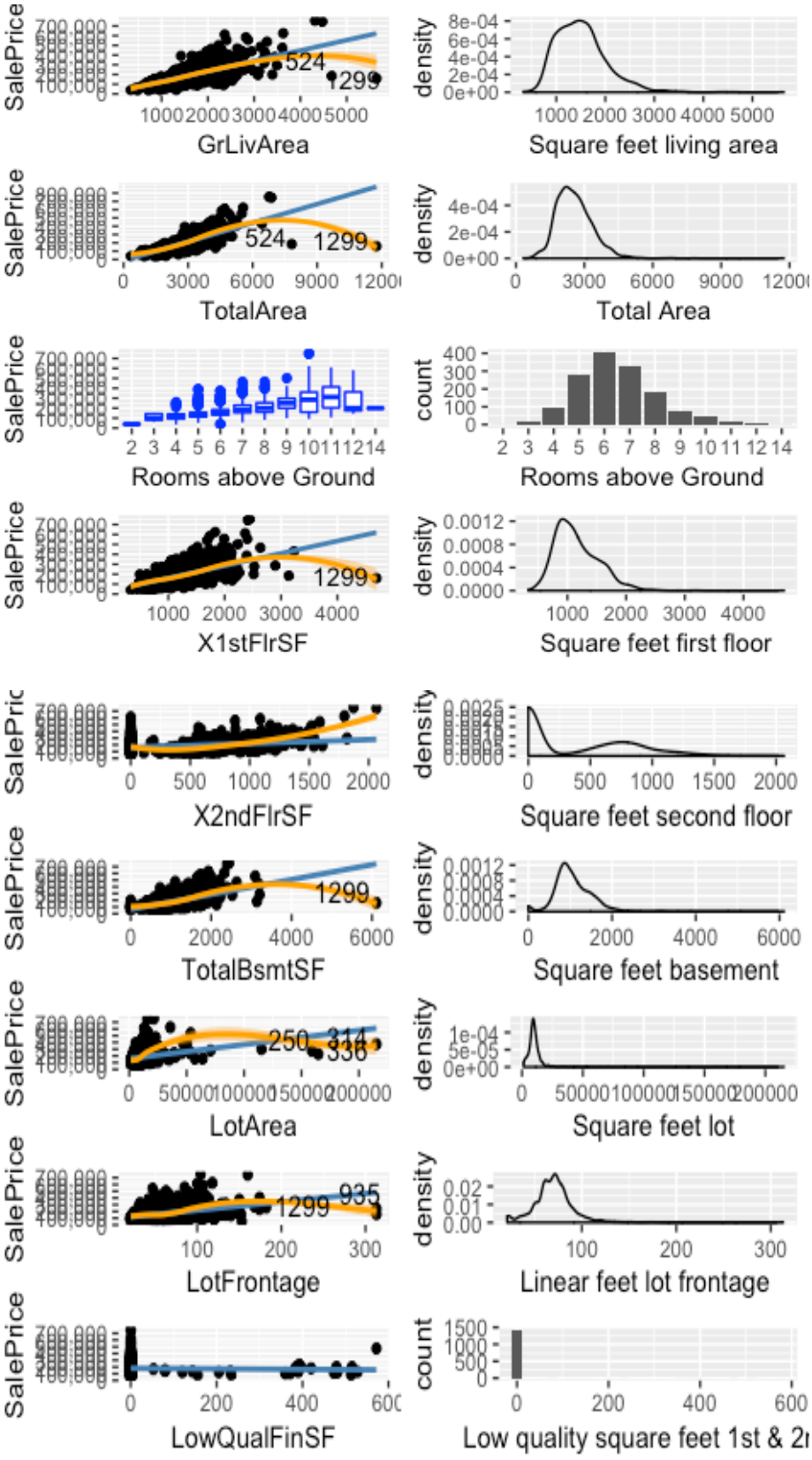



```
ggpairs(table, effecive_var[c(1,6:11)], lower = list(continuous = lm.plt))
```



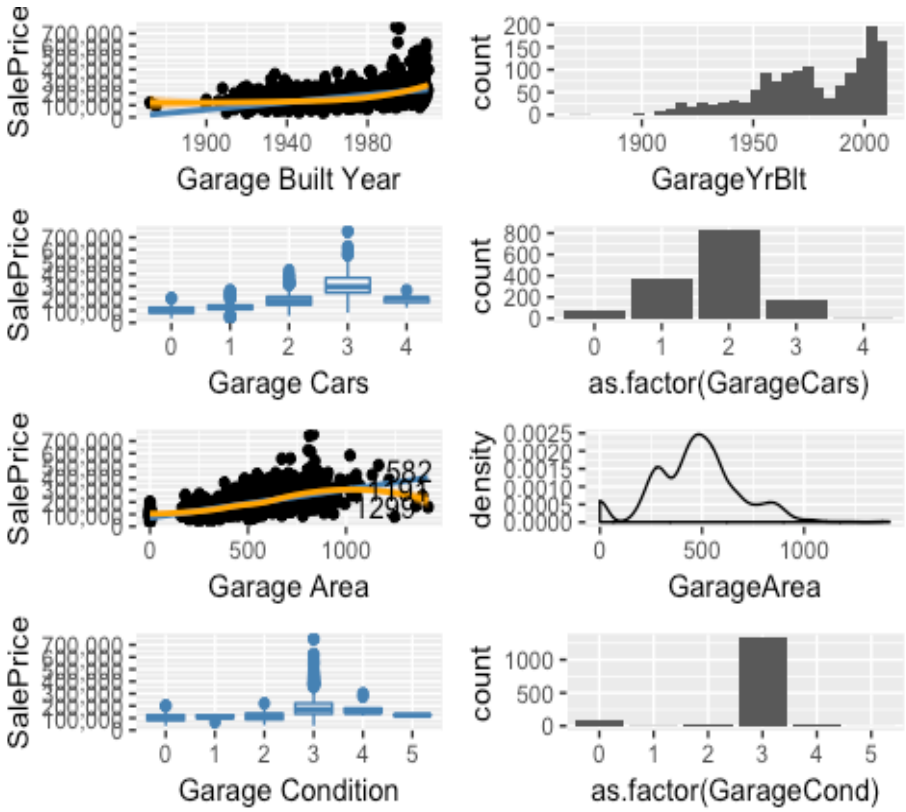
It shows that the variables are not normal. So, I need to normalize them later.

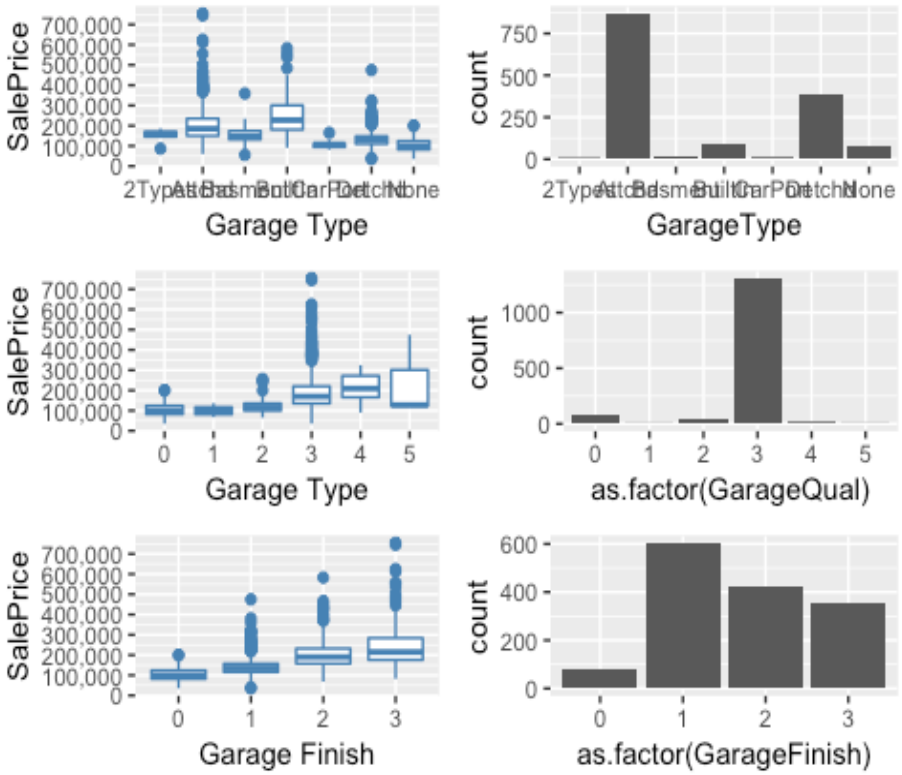
4.2 Relation between SalePrice and variables related to area variables



Clearly, there are two outliers: 524 and 1299. So, I will delete them later. And I notice that some variables are high correlated and I will create a new variable called total area which is sum of = TotalBsmtSF, X1stFlrSF, X2ndFlrSF, which can well include information of three variables. And I notice that GrLivArea = X1stFlrSF+X2ndFlrSF+LowQualFinSF. So, I will keep GrLivArea and delete other three variables.

4.3 Relation between SalePrice and variables related to Garage variables

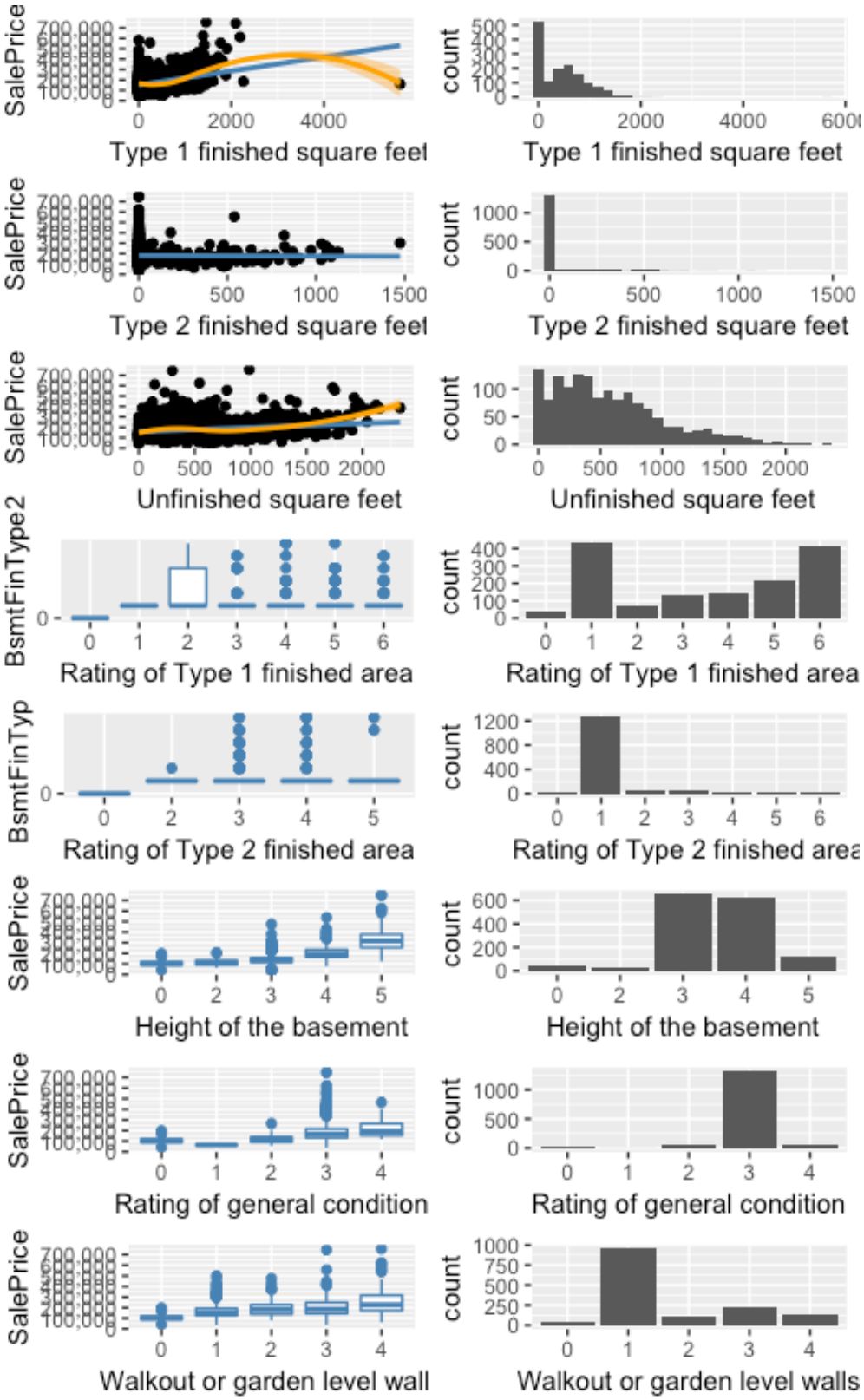




Based on the correlation matrix, it can be concluded that, several garage variables have high correlation between SalaPrice. However, there are multicollinearity among them: GarageCars & GarageArea and GarageCond & GarageQual. Also, seven variables of garage variables in regression would be too many. In my opinion, GarageCars, GarageType and GarageCond would be enough.

4.4 Relation between SalePrice and variables related to Basement variables

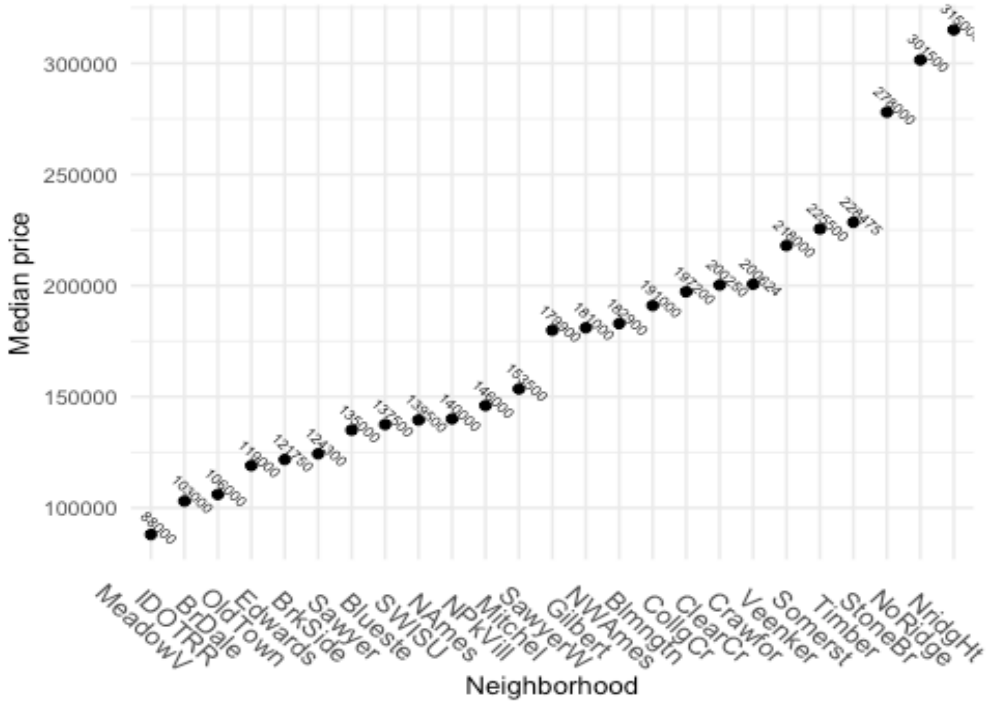
Like Garage variables, multiple variables are important in the correlation matrix. And there is multicollinearity issue. Also, 11 variables of basement would be too many for regression.



Firstly, it seems that there is an outlier. And I will check it later. And it seemed as if the Total Basement Surface in square feet (TotalBsmtSF) is further broken down into finished areas (2 if

more than one type of finish), and unfinished area. I did a check between the correlation of total of those 3 variables, and TotalBsmtSF. And I find that the correlation is one. So, I will eliminate the not important variables: finished area and unfinished area.

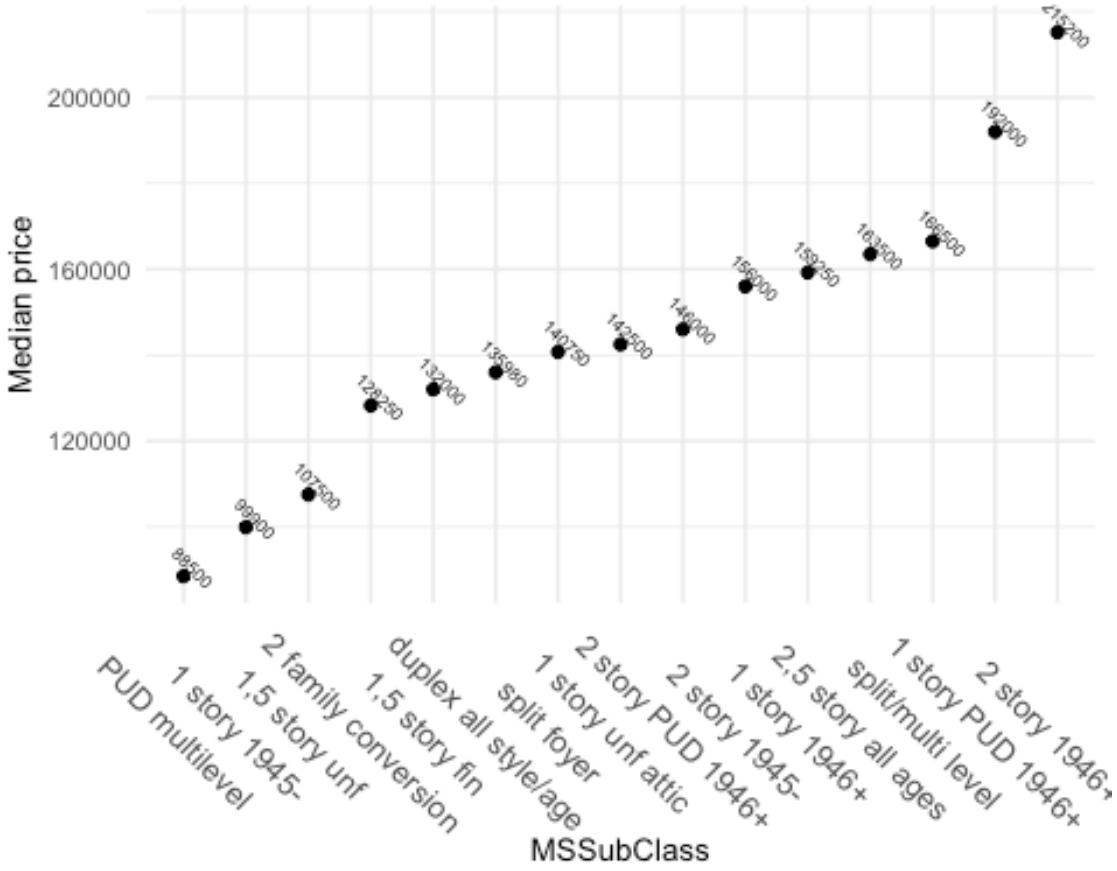
4.5 Neighborhoods variables



Clearly, there is cluster. So, I revalue the neighbor variable based on cluster.

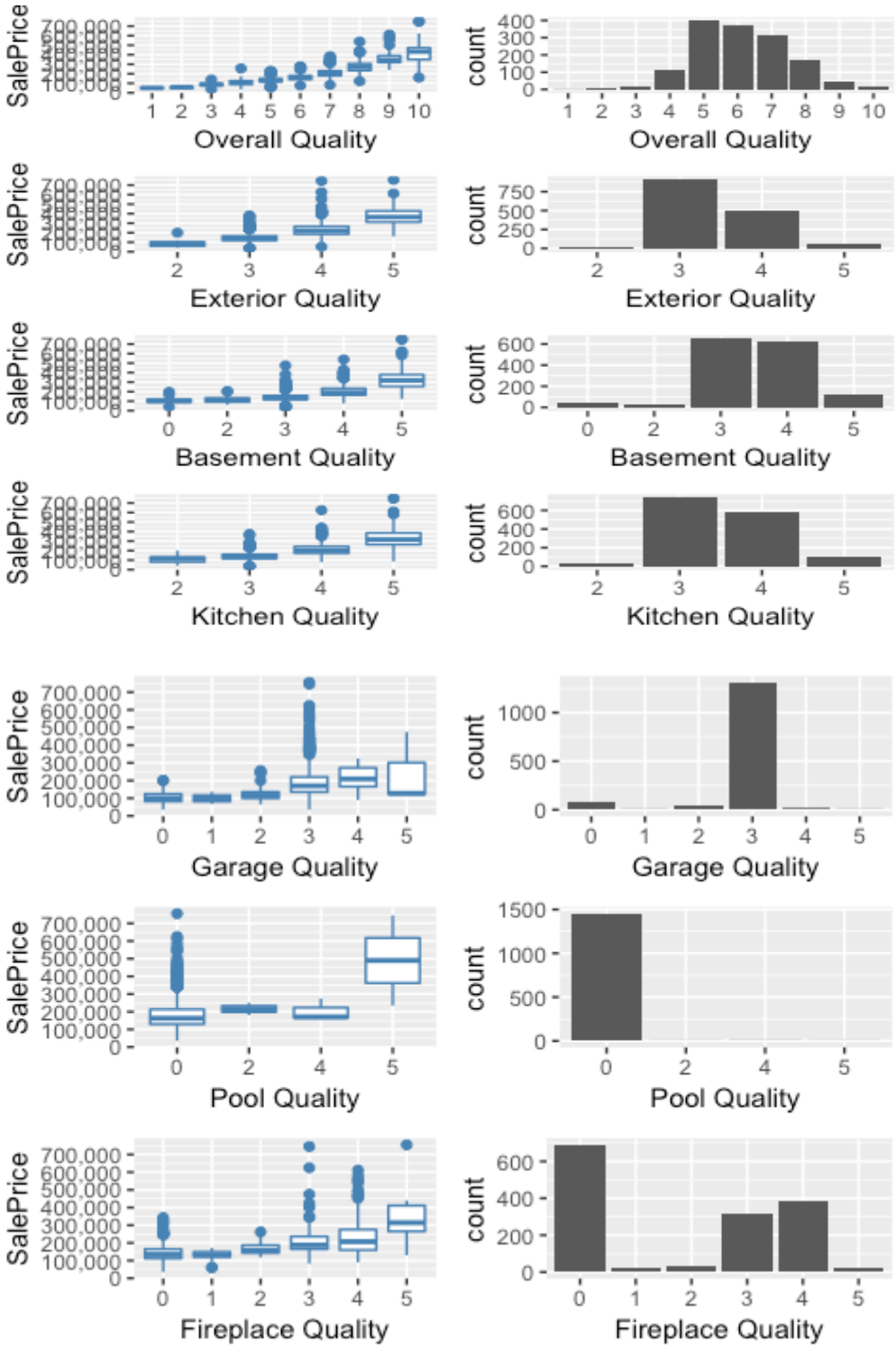
```
nbrh.map <- c('MeadowV' = 0, 'IDOTRR' = 1, 'BrDale' = 1, 'OldTown' = 2, 'Edwa
rds' = 2,
              'BrkSide' = 2, 'Sawyer' = 3,'Blueste' = 3, 'SWISU' = 3, 'NAmes'
= 3, 'NPkVill' = 3, 'Mitchel' = 4,
              'SawyerW' = 4, 'Gilbert' = 5, 'NWAmes' = 5, 'Blmngtn' = 5, 'Coll
gCr' = 6, 'ClearCr' = 6,
              'Crawfor' = 6, 'Veenker' = 6, 'Somerst' = 7, 'Timber' = 7, 'Ston
eBr' = 8, 'NoRidge' = 9,
              'NridgHt' = 10)

table$Neighclass<-as.integer(revalue(table$Neighborhood,nbrh.map))
```

4.6 MSSubClass



Clearly, it also has cluster.

4.7 Relation between SalePrice and variables related to Quality variables



Overall Quality is very important, and also more granular than the other variables. External Quality is also important, but has a high correlation with Overall Quality (0.73). Kitchen Quality also seems one to keep, as all houses have a kitchen and there is a variance with some substance. Garage Quality does not seem to distinguish much, as the majority of garages have Q3. Fireplace

Quality is in the list of high correlations, and in the important variables list. The PoolQC is just very sparse (the 13 pools cannot even be seen on this scale). I will look at creating a 'has pool' variable later on.

5. Feature engineering

In this section I will create some variables and delete some useless variables to decrease multicollinearity.

5.1 Total Area

As did in 4.2 TotalArea = TotalBsmtSF + X1stFlrSF + X2ndFlrSF

```
table$TotalArea<-table$TotalBsmtSF+table$X1stFlrSF+table$X2ndFlrSF
```

5.2 Total Bathroom

TotBathrooms = FullBath + (HalfBath*0.5) + BsmtFullBath + (BsmtHalfBath*0.5)

```
table$TotBathrooms <- table$FullBath + (table$HalfBath*0.5) + table$BsmtFullB
ath + (table$BsmtHalfBath*0.5)

cor(table$TotBathrooms,table$SalePrice)

## [1] 0.6317311
```

5.3 Consolidating Porch variables

TotalPorchSF = OpenPorchSF + EnclosedPorch + X3SsnPorch + ScreenPorch

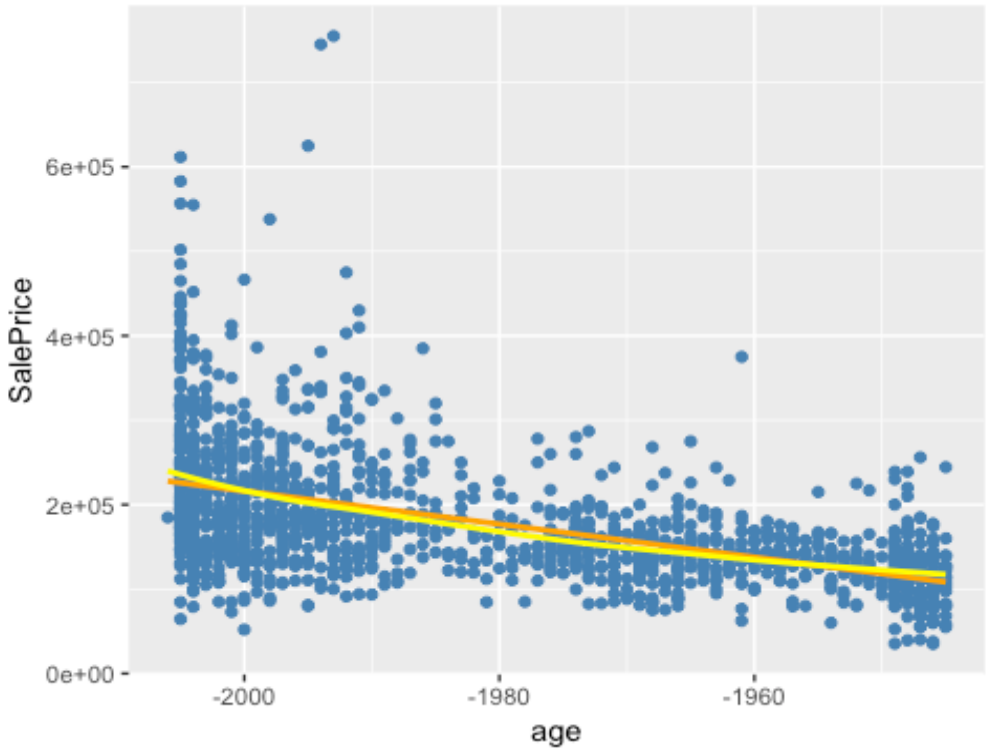
```
table$TotalPorchSF <- table$OpenPorchSF + table$EnclosedPorch + table$X3SsnPo
rch + table$ScreenPorch
cor(table$SalePrice, table$TotalPorchSF)

## [1] 0.1957389
```

5.4 House Age

Age = YrSold – YearRemodAdd

```
table$age<-as.numeric(table$YrSold)-table$YearRemodAdd
ggplot(data=table[!is.na(table$SalePrice),], aes(x=age, y=SalePrice))+
        geom_point(col='steelblue') +
        geom_smooth(method = "lm", se=FALSE, color="orange", aes(group=1)) +
        geom_smooth(method = "loess", se=FALSE, color="yellow", aes(group=1))

cor(table$SalePrice[!is.na(table$SalePrice)], table$age[!is.na(table$SalePric
e)])
```

```
## [1] -0.5090787
```



5.5 The house is remodeled or not

There are three variables that are relevant with regards to the age of a house: YearBlt, YearRemodAdd and YearSold. As introduced before I use YearRemodeled and YearSold to determine the age of house. And then I will introduce a remod variables to represent if the house was remodeled or not.

```
table$remod <- ifelse(table$YearBuilt==table$YearRemodAdd, 0, 1)
```

6. Preparing data for modeling

6.1 Avoiding outliers

As I mentioned before, there are two unreasonable record: 1299 and 524. I remove these 2 records.

```
table<-table[-c(1299,524)]
```

6.2 Dropping high correlated variables

As analyzed before, some variables contain repeated information. For example, GarageCars and GarageArea have a high correlation of 0.88. And correlation between GarageArea and SalePrice

is 0.62. However, correlation between GarageCars and SalePrice is 0.64. So, I drop GarageArea that is with lower correlation with SalePrice.

```
dropve<-c('YearRemodAdd', 'GarageYrBlt', 'YearBuilt', 'GarageArea', 'GarageFinish', 'GarageCond', 'TotalBsmtSF', 'TotalRmsAbvGrd', 'BsmtFinSF1','OpenPorchSF','EnclosedPorch', 'X3SsnPorch',  'ScreenPorch','FullBath', 'HalfBath','BsmtFullBath', 'BsmtHalfBath','LowQualFinSF', 'X1stFlrSF', 'X2ndFlrSF','LowQualFinSF','BsmtCond','GrLivArea','Neighborhood' )
table<-table[,!(names(table) %in% dropve)]

#num_copy<-copy(num_var)
num_var_new<-num_var_name[!(num_var_name %in% c('MSSubClass', 'MoSold', 'YrSold', 'SalePrice', 'OverallQual', 'OverallCond')) & !(num_var_name %in% dropve) ]
num_var_new <- append(num_var_new, c('age', 'TotalPorchSF', 'TotBathrooms'))
length(num_var_new)

## [1] 16
```

After dropping some variables, there are 16 numeric variables.

6.3 Skewness and normalizing of numeric variable

I check skewness for each variable and take a log transformation for the variables whose skewness are greater than 0.75. Then the final explanatory variables for model is given by com_var.

```
df_num<-table[,names(table) %in% num_var_new]

for(i in 1:ncol(df_num)){
        if (abs(skew(df_num[,i]))>0.75){
                df_num[,i] <- log(df_num[,i] +1)
        }
}

prenum <- preProcess(df_num, method=c("center", "scale"))
print(prenum)

## Created from 1460 samples and 16 variables
##
## Pre-processing:
##   - centered (16)
##   - ignored (0)
##   - scaled (16)

df_nor <- predict(prenum, df_num)

df_fac <- table[, !(names(table) %in% num_var_new)]
df_fac <- df_fac[, names(df_fac) != 'SalePrice']
```

```
com_var<-cbind(df_nor, df_fac)
sum(is.na(com_var))
```

```
## [1] 0
```

```
names(com_var)
```

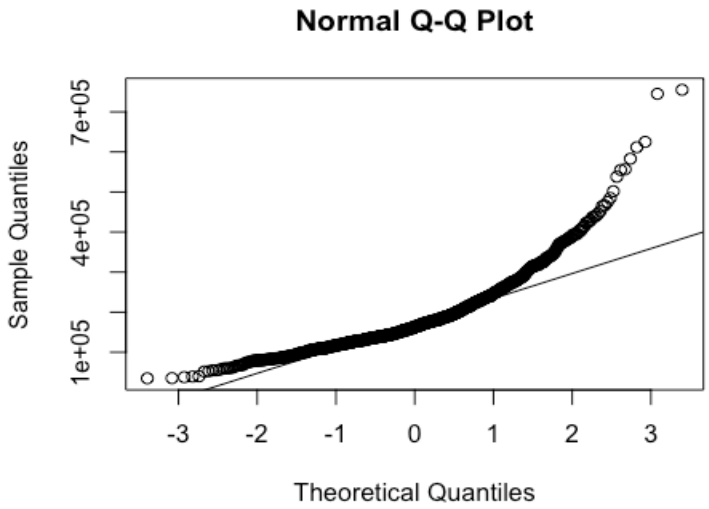
```
##  [1] "LotFrontage"   "LotArea"       "MasVnrArea"    "BsmtFinSF2"
##  [5] "BsmtUnfSF"     "BedroomAbvGr"  "KitchenAbvGr"  "TotRmsAbvGrd"
##  [9] "Fireplaces"    "GarageCars"    "WoodDeckSF"    "PoolArea"
## [13] "MiscVal"       "TotBathrooms"  "TotalPorchSF"  "age"
## [17] "MSSubClass"    "MSZoning"      "Street"        "Alley"
## [21] "LotShape"      "LandContour"   "LotConfig"     "LandSlope"
## [25] "Condition1"    "Condition2"    "BldgType"      "HouseStyle"
## [29] "OverallQual"   "OverallCond"   "RoofStyle"     "RoofMatl"
## [33] "Exterior1st"   "Exterior2nd"   "MasVnrType"    "ExterQual"
## [37] "ExterCond"     "Foundation"    "BsmtQual"      "BsmtExposure"
## [41] "BsmtFinType1"  "BsmtFinType2"  "Heating"       "HeatingQC"
## [45] "CentralAir"    "Electrical"    "KitchenQual"   "Functional"
## [49] "FireplaceQu"   "GarageType"    "GarageQual"    "PavedDrive"
## [53] "PoolQC"        "Fence"         "MiscFeature"   "MoSold"
## [57] "YrSold"        "SaleType"      "SaleCondition" "TotalArea"
## [61] "Neighclass"    "newornot"      "remod"
```

6.4 Skewness and normalizing of response variables

```
skew(table$SalePrice)
```

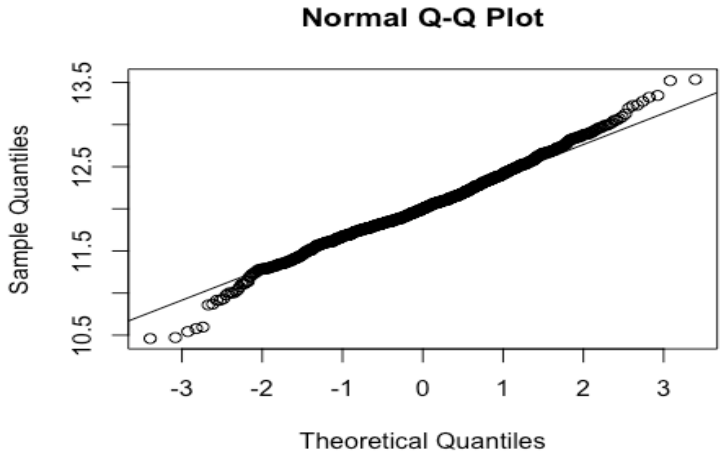
```
## [1] 1.879009
```

```
qqnorm(table$SalePrice)
qqline(table$SalePrice)
```



The skeness is 1.88 which indicates a right skew. And the Q-Q plot shows that it is not noemally distributed. So, I take a log transformation to fix it.

```
table$SalePrice <- log(table$SalePrice)
qqnorm(table$SalePrice)
qqline(table$SalePrice)
```



7. Modeling

7.1 Random Forest – with IncNodePurity and IncMSE

First I use Random Forest model to choose important parameters with IncNodePurity.

```
set.seed(10)
model1.rf<-randomForest(x=com_var, y=table$SalePrice, ntree=100,importance=TR
UE)
model1.rf<-importance(model1.rf,2)
#imp_df<-data.frame(Variables = row.names(imp_rf), MSE = imp_rf[,1])
#imp_df<-imp_df[order(imp_df$MSE, decreasing = TRUE),]

sort_var<-model1.rf[order(model1.rf,decreasing = TRUE),]
sort_var

##     TotalArea   OverallQual     ExterQual  TotBathrooms    GarageCars
##  63.097723153  61.191589737  13.892892053  11.026563882   8.907164275
##      BsmtQual    MSSubClass   KitchenQual    Neighclass       LotArea
##   7.283765293   5.967896525   5.508595205   5.153016156   3.905745185
##       MoSold    GarageType           age   Exterior2nd    Fireplaces
##   3.602235859   3.406652615   3.215884448   2.356876361   2.283398871
##    CentralAir   OverallCond   Exterior1st   FireplaceQu     BsmtUnfSF
##   2.228073133   2.100014609   2.015797600   1.958450768   1.926814018
```

```
##   LotFrontage      MSZoning  TotRmsAbvGrd  TotalPorchSF  BsmtFinType1
##   1.852608668   1.570702629   1.429703999   1.127872565   0.949140941
##        YrSold    GarageQual    MasVnrArea SaleCondition    WoodDeckSF
##   0.914975243   0.907911515   0.890316660   0.748258331   0.697210359
##    HouseStyle  BedroomAbvGr    Foundation    PavedDrive   LandContour
##   0.640158764   0.636147324   0.590013674   0.564035116   0.547799382
##  BsmtExposure     HeatingQC      LotShape     ExterCond    Condition1
##   0.480622456   0.455440171   0.417302412   0.402398459   0.388507534
##    Functional  KitchenAbvGr     LotConfig         Fence     RoofStyle
##   0.376715630   0.334034004   0.331869908   0.321015706   0.308987848
##      BldgType    Electrical      SaleType         Alley    MasVnrType
##   0.304892071   0.238335899   0.223634183   0.201740811   0.162080944
##  BsmtFinType2      RoofMatl     LandSlope      newornot         remod
##   0.154370333   0.148979133   0.147551579   0.110941188   0.103947128
##       Heating    BsmtFinSF2        PoolQC       MiscVal   MiscFeature
##   0.100309773   0.093117510   0.060714949   0.043475749   0.029606437
##    Condition2        Street      PoolArea
##   0.015513704   0.014327174   0.005782243
```

I pick up TotalArea, OverallQual, Neighclass, TotBathrooms, KitchenQual, GarageCars, BsmtQual, ExterQual, MSSubClass, age, MoSold.
Then I try to use IncMSE as criteria to pick important features.

```
set.seed(2)
model2.rf<-randomForest(x=com_var, y=table$SalePrice, ntree=100,importance=TR
UE)
imp_rf<-importance(model2.rf,2)
imp_df<-data.frame(Variables = row.names(imp_rf), MSE = imp_rf[,1])
imp_df<-imp_df[order(imp_df$MSE, decreasing = TRUE),]

imp_df
```

```
##                   Variables          MSE
## TotalArea         TotalArea 62.957075352
## OverallQual     OverallQual 43.753562642
## KitchenQual     KitchenQual 18.634390919
## ExterQual         ExterQual 14.260980105
## TotBathrooms   TotBathrooms 11.737241001
## GarageCars       GarageCars 10.574916640
## BsmtQual           BsmtQual  8.050141182
## MSSubClass       MSSubClass  5.673290287
## Neighclass       Neighclass  4.550017037
## LotArea             LotArea  4.034130620
## age                     age  3.614055355
## MoSold               MoSold  3.555079090
## CentralAir       CentralAir  3.111178120
## FireplaceQu     FireplaceQu  2.880134863
## Exterior2nd     Exterior2nd  2.488548442
## GarageType       GarageType  2.252580451
## Exterior1st     Exterior1st  2.225046586
```

```
## TotRmsAbvGrd   TotRmsAbvGrd  2.052824831
## OverallCond     OverallCond  1.857504981
## Fireplaces       Fireplaces  1.841049380
## LotFrontage     LotFrontage  1.837549737
## BsmtUnfSF         BsmtUnfSF  1.824532483
## GarageQual       GarageQual  1.423707184
## TotalPorchSF   TotalPorchSF  1.378397602
## MSZoning           MSZoning  1.154771485
## HeatingQC         HeatingQC  0.958702736
## YrSold               YrSold  0.933968410
## BsmtFinType1   BsmtFinType1  0.929525322
## BedroomAbvGr   BedroomAbvGr  0.882787996
## MasVnrArea       MasVnrArea  0.871459470
## SaleCondition SaleCondition  0.732757798
## WoodDeckSF       WoodDeckSF  0.674605072
## HouseStyle       HouseStyle  0.617422060
## Foundation       Foundation  0.545686058
## BsmtExposure   BsmtExposure  0.537903863
## LandContour     LandContour  0.528209591
## Fence                 Fence  0.449916238
## PavedDrive       PavedDrive  0.435378881
## Functional       Functional  0.422501640
## LotShape           LotShape  0.384024220
## Condition1       Condition1  0.339108218
## ExterCond         ExterCond  0.330991288
## RoofStyle         RoofStyle  0.310563828
## LotConfig         LotConfig  0.290122530
## SaleType           SaleType  0.239314590
## BldgType           BldgType  0.239251235
## MasVnrType       MasVnrType  0.214225114
## Alley                 Alley  0.181734663
## LandSlope         LandSlope  0.169527496
## Electrical       Electrical  0.162278285
## BsmtFinType2   BsmtFinType2  0.136131533
## remod                 remod  0.127952621
## BsmtFinSF2       BsmtFinSF2  0.122204471
## newornot           newornot  0.112295263
## KitchenAbvGr   KitchenAbvGr  0.105046920
## Heating             Heating  0.097797466
## PoolArea           PoolArea  0.085509124
## RoofMatl           RoofMatl  0.076385676
## Condition2       Condition2  0.070420938
## MiscVal             MiscVal  0.065755459
## PoolQC               PoolQC  0.062638095
## MiscFeature     MiscFeature  0.038126116
## Street               Street  0.002077188
```

As can be seen, the result is consistent with before.

```
rf_model <- train(x=com_var2, y=table$SalePrice[!is.na(table$SalePrice)], dat
a=table,method="rf",metric="RMSE",
                  maximize=FALSE,trControl=trainControl(method="repeatedcv
",number=5),
                  tuneGrid=expand.grid(mtry = c(5)), importance = T, allow
Parallel = T, prox = T)

rf_model$results

##   mtry  RMSE Rsquared   MAE  RMSESD RsquaredSD   MAESD
## 1    5 0.157    0.867 0.104 0.00903     0.0355 0.00272
```

The result shows that RMSE of random forest method is 0.157 and Rsquared is 0.867.

7.2 GAM

Then I use the parameters picked by random forest to fit the smooth function – gam.

```
lmgam <- gam(table$SalePrice ~ TotalArea + OverallQual+ Neighclass+ TotBathro
oms+ KitchenQual+ GarageCars+ BsmtQual+ ExterQual+ MSSubClass+ MoSold+ age,
             data=table)
summary(lmgam)

##
## Family: gaussian
## Link function: identity
##
## Formula:
## table$SalePrice ~ TotalArea + OverallQual + Neighclass + TotBathrooms +
##     KitchenQual + GarageCars + BsmtQual + ExterQual + MSSubClass +
##     MoSold + age
##
## Parametric coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    7.872e+00  5.400e-01  14.578  < 2e-16 ***
## TotalArea                      1.312e-04  8.139e-06  16.123  < 2e-16 ***
## OverallQual                    8.280e-02  5.657e-03  14.637  < 2e-16 ***
## Neighclass                     8.940e-03  1.768e-03   5.056 4.83e-07 ***
## TotBathrooms                   7.845e-02  7.841e-03  10.005  < 2e-16 ***
## KitchenQual                    3.775e-02  1.006e-02   3.752 0.000182 ***
## GarageCars                     7.444e-02  7.497e-03   9.929  < 2e-16 ***
## BsmtQual                       1.375e-02  6.816e-03   2.017 0.043890 *
## ExterQual                      1.257e-02  1.232e-02   1.020 0.308088
## MSSubClass1 story 1945-       -1.809e-01  2.135e-02  -8.475  < 2e-16 ***
## MSSubClass1 story unf attic    9.161e-03  7.891e-02   0.116 0.907588
## MSSubClass1,5 story unf       -1.307e-01  4.648e-02  -2.811 0.005003 **
## MSSubClass1,5 story fin       -3.641e-02  1.544e-02  -2.358 0.018500 *
## MSSubClass2 story 1946+        5.964e-03  1.268e-02   0.470 0.638236
## MSSubClass2 story 1945-       -2.140e-02  2.181e-02  -0.981 0.326705
## MSSubClass2,5 story all ages  -5.225e-02  4.031e-02  -1.296 0.195053
```

```
## MSSubClasssplit/multi level     3.074e-02  2.209e-02   1.391 0.164347
## MSSubClasssplit foyer          -6.088e-03  3.616e-02  -0.168 0.866321
## MSSubClassduplex all style/age -1.259e-01  2.435e-02  -5.170 2.68e-07 ***
## MSSubClass1 story PUD 1946+    -7.140e-02  1.912e-02  -3.735 0.000195 ***
## MSSubClass2 story PUD 1946+    -2.120e-01  2.187e-02  -9.691  < 2e-16 ***
## MSSubClassPUD multilevel       -2.112e-01  5.146e-02  -4.104 4.28e-05 ***
## MSSubClass2 family conversion  -9.618e-02  3.022e-02  -3.183 0.001490 **
## MoSold2                         1.956e-02  3.011e-02   0.649 0.516166
## MoSold3                         4.492e-02  2.566e-02   1.751 0.080240 .
## MoSold4                         3.729e-02  2.455e-02   1.519 0.129105
## MoSold5                         5.325e-02  2.344e-02   2.271 0.023268 *
## MoSold6                         6.400e-02  2.298e-02   2.785 0.005427 **
## MoSold7                         6.003e-02  2.315e-02   2.593 0.009620 **
## MoSold8                         4.824e-02  2.509e-02   1.923 0.054704 .
## MoSold9                         3.072e-02  2.869e-02   1.071 0.284478
## MoSold10                        2.902e-03  2.671e-02   0.109 0.913494
## MoSold11                        3.425e-02  2.721e-02   1.259 0.208234
## MoSold12                        4.277e-02  2.915e-02   1.467 0.142640
## age                            -1.377e-03  2.825e-04  -4.875 1.21e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## R-sq.(adj) =  0.847   Deviance explained = 85.1%
## GCV = 0.024981  Scale est. = 0.024382  n = 1460
```

Obviously, there are some variables are not significant. In gam model the Rsquared is 0.847.

7.3 Natural Cubic Spline and cross validation

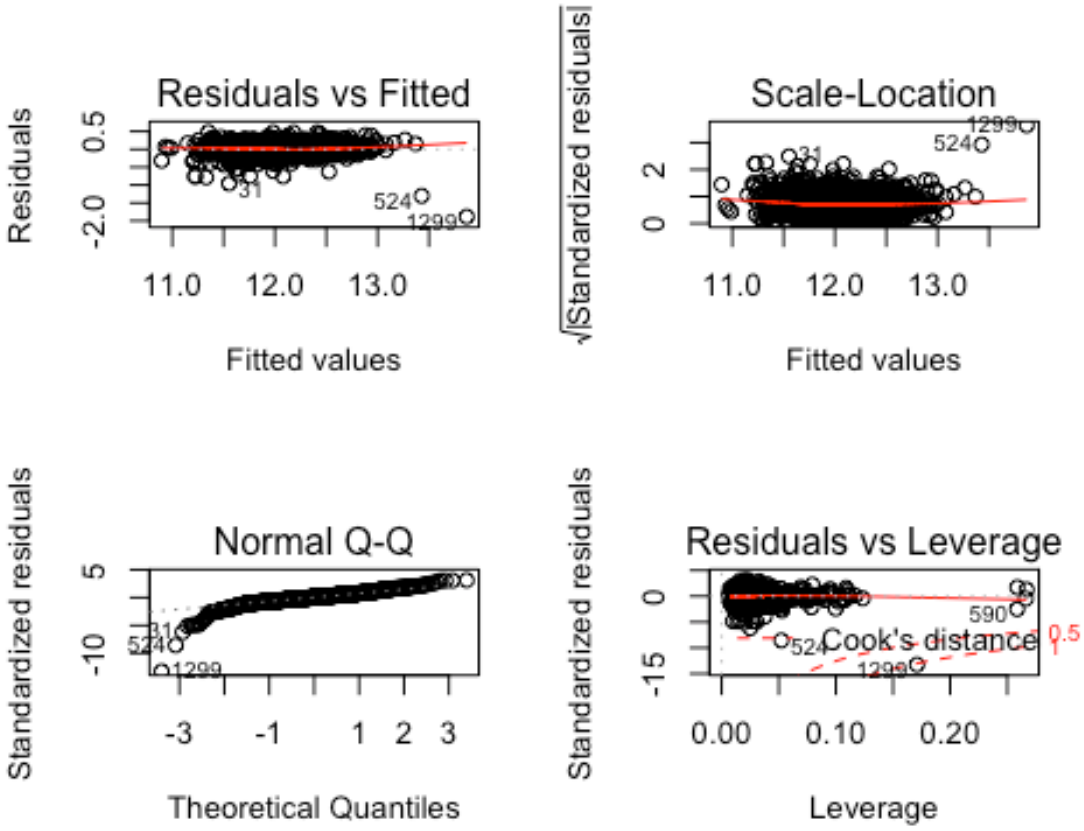
I fit natural cubic spline with variables picked by random forest.

```
ns_model<-lm(table$SalePrice~ ns(TotalArea) + OverallQual+ Neighclass+ ns(Tot
Bathrooms)+ KitchenQual+ ns(GarageCars) + BsmtQual+ ExterQual+ MSSubClass+ Mo
Sold+ ns(age), data=table)
summary(ns_model)
```

```
##
## Call:
## lm(formula = table$SalePrice ~ ns(TotalArea) + OverallQual +
##     Neighclass + ns(TotBathrooms) + KitchenQual + ns(GarageCars) +
##     BsmtQual + ExterQual + MSSubClass + MoSold + ns(age), data = table)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.88460 -0.06924  0.00630  0.07915  0.47972
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)                    10.757178   0.047380 227.039  < 2e-16 ***
## ns(TotalArea)                   1.868671   0.115899  16.123  < 2e-16 ***
## OverallQual                     0.082798   0.005657  14.637  < 2e-16 ***
## Neighclass                      0.008940   0.001768   5.056 4.83e-07 ***
## ns(TotBathrooms)                0.489196   0.048895  10.005  < 2e-16 ***
## KitchenQual                     0.037752   0.010061   3.752 0.000182 ***
## ns(GarageCars)                  0.371354   0.037402   9.929  < 2e-16 ***
## BsmtQual                        0.013747   0.006816   2.017 0.043890 *
## ExterQual                       0.012566   0.012324   1.020 0.308088
## MSSubClass1 story 1945-        -0.180940   0.021350  -8.475  < 2e-16 ***
## MSSubClass1 story unf attic     0.009161   0.078907   0.116 0.907588
## MSSubClass1,5 story unf        -0.130659   0.046478  -2.811 0.005003 **
## MSSubClass1,5 story fin        -0.036405   0.015438  -2.358 0.018500 *
## MSSubClass2 story 1946+         0.005964   0.012683   0.470 0.638236
## MSSubClass2 story 1945-        -0.021395   0.021807  -0.981 0.326705
## MSSubClass2,5 story all ages   -0.052253   0.040307  -1.296 0.195053
## MSSubClasssplit/multi level     0.030738   0.022093   1.391 0.164347
## MSSubClasssplit foyer          -0.006088   0.036161  -0.168 0.866321
## MSSubClassduplex all style/age -0.125872   0.024348  -5.170 2.68e-07 ***
## MSSubClass1 story PUD 1946+    -0.071405   0.019120  -3.735 0.000195 ***
## MSSubClass2 story PUD 1946+    -0.211989   0.021874  -9.691  < 2e-16 ***
## MSSubClassPUD multilevel       -0.211222   0.051461  -4.104 4.28e-05 ***
## MSSubClass2 family conversion  -0.096179   0.030217  -3.183 0.001490 **
## MoSold2                         0.019555   0.030112   0.649 0.516166
## MoSold3                         0.044920   0.025661   1.751 0.080240 .
## MoSold4                         0.037285   0.024554   1.519 0.129105
## MoSold5                         0.053253   0.023444   2.271 0.023268 *
## MoSold6                         0.063999   0.022982   2.785 0.005427 **
## MoSold7                         0.060032   0.023154   2.593 0.009620 **
## MoSold8                         0.048235   0.025086   1.923 0.054704 .
## MoSold9                         0.030718   0.028689   1.071 0.284478
## MoSold10                        0.002902   0.026713   0.109 0.913494
## MoSold11                        0.034252   0.027206   1.259 0.208234
## MoSold12                        0.042765   0.029155   1.467 0.142640
## ns(age)                        -0.104783   0.021494  -4.875 1.21e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1561 on 1425 degrees of freedom
## Multiple R-squared:  0.8508, Adjusted R-squared:  0.8472
## F-statistic: 238.9 on 34 and 1425 DF,  p-value: < 2.2e-16
```

```r
layout(matrix(c(1,2,3,4),2,2)) # optional 4 graphs/page
plot(ns_model)
```

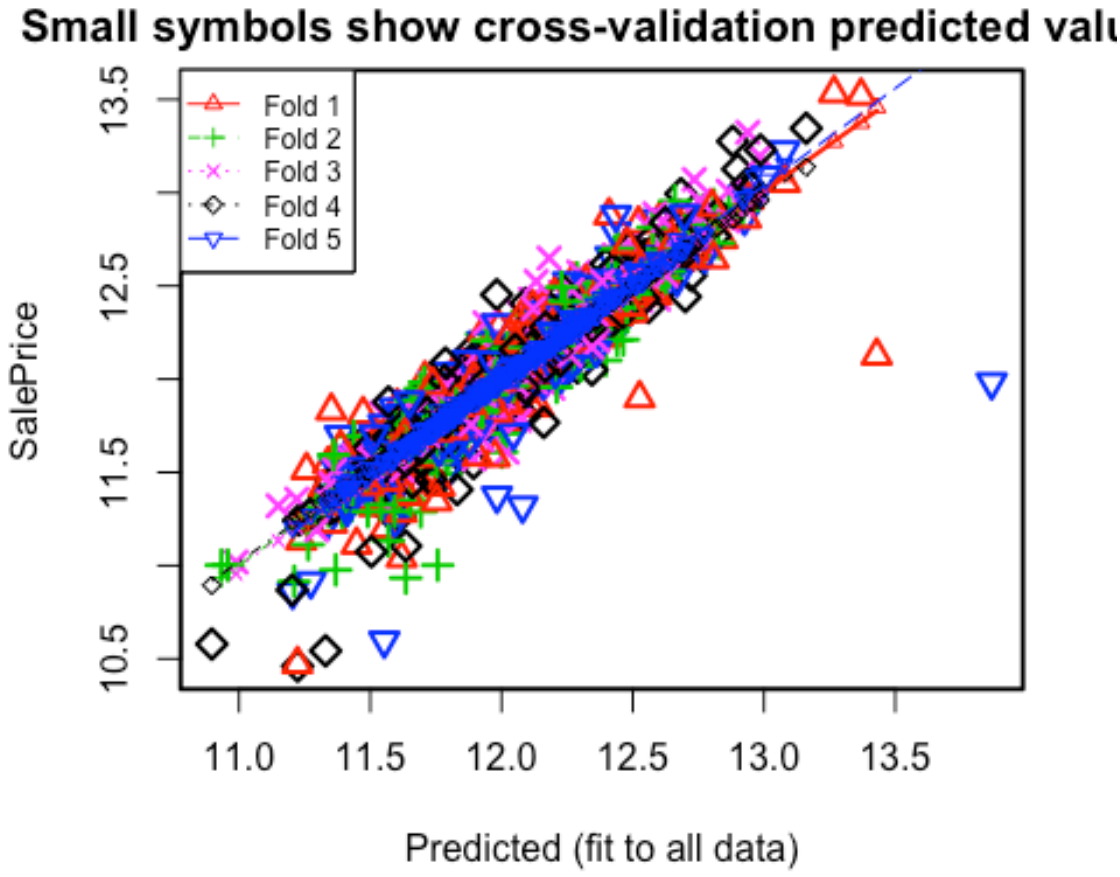


The R-squared is 0.8472. And I check the plot of residuls, which show that the residuls distribute as normal distribution, which is a good result.

Next I try the lm model with cross validation with k=5. The predicted value for SalePrice is given below. And overall sum of squares is 0.026, which is pretty small. What is worth noting is that in natral cubic spline model I can use steAIC to choose model (choose variables in lm). And the result is given after the result of cross validation. The result shows that I can use TotalArea, OverallQual, Neighclass, TotBathrooms, KitchenQual, GarageCars, MSSubClass,age to fit the natural cubic spline model which has minimum AIC value.

```
form<-SalePrice~ (ns(TotalArea) + OverallQual+ Neighclass+ ns(TotBathrooms)+
KitchenQual+ ns(GarageCars) + BsmtQual+ ExterQual+ MSSubClass+ MoSold+ ns(ag
e))
g<-lm(form,data=table)
a<-cv.lm(data =table,form.lm=form,m=5)

## Analysis of Variance Table
##
## Response: SalePrice
##                    Df Sum Sq Mean Sq F value  Pr(>F)
## ns(TotalArea)       1  140.7   140.7 5768.87 < 2e-16 ***
```

```
## OverallQual          1   37.3    37.3 1529.97 < 2e-16 ***
## Neighclass           1    1.2     1.2   49.06 3.8e-12 ***
## ns(TotBathrooms)     1    6.1     6.1  251.37 < 2e-16 ***
## KitchenQual          1    2.2     2.2   92.25 < 2e-16 ***
## ns(GarageCars)       1    3.9     3.9  160.06 < 2e-16 ***
## BsmtQual             1    0.3     0.3   10.40  0.0013 **
## ExterQual            1    0.0     0.0    1.65  0.1990
## MSSubClass          14    5.2     0.4   15.20 < 2e-16 ***
## MoSold              11    0.6     0.1    2.08  0.0191 *
## ns(age)              1    0.6     0.6   23.77 1.2e-06 ***
## Residuals         1425   34.7     0.0
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```




```
tep <- stepAIC(ns_model, direction="both")

## Start:  AIC=-5388
## table$SalePrice ~ ns(TotalArea) + OverallQual + Neighclass +
##     ns(TotBathrooms) + KitchenQual + ns(GarageCars) + BsmtQual +
##     ExterQual + MSSubClass + MoSold + ns(age)
##
##                    Df Sum of Sq  RSS   AIC
## - MoSold           11      0.49 35.2 -5389
```

```
## - ExterQual         1      0.03 34.8 -5389
## <none>                          34.7 -5388
## - BsmtQual          1      0.10 34.8 -5386
## - KitchenQual       1      0.34 35.1 -5375
## - ns(age)           1      0.58 35.3 -5366
## - Neighclass        1      0.62 35.4 -5364
## - ns(GarageCars)    1      2.40 37.1 -5292
## - ns(TotBathrooms)  1      2.44 37.2 -5291
## - MSSubClass       14      4.99 39.7 -5220
## - OverallQual       1      5.22 40.0 -5185
## - ns(TotalArea)     1      6.34 41.1 -5145
##
## Step:  AIC=-5389
## table$SalePrice ~ ns(TotalArea) + OverallQual + Neighclass +
##     ns(TotBathrooms) + KitchenQual + ns(GarageCars) + BsmtQual +
##     ExterQual + MSSubClass + ns(age)
##
##                    Df Sum of Sq  RSS   AIC
## - ExterQual         1      0.03 35.3 -5390
## <none>                          35.2 -5389
## - BsmtQual          1      0.08 35.3 -5388
## + MoSold           11      0.49 34.7 -5388
## - KitchenQual       1      0.32 35.6 -5378
## - ns(age)           1      0.65 35.9 -5365
## - Neighclass        1      0.65 35.9 -5365
## - ns(TotBathrooms)  1      2.44 37.7 -5293
## - ns(GarageCars)    1      2.47 37.7 -5292
## - MSSubClass       14      4.95 40.2 -5226
## - OverallQual       1      5.23 40.5 -5189
## - ns(TotalArea)     1      6.34 41.6 -5150
##
## Step:  AIC=-5390
## table$SalePrice ~ ns(TotalArea) + OverallQual + Neighclass +
##     ns(TotBathrooms) + KitchenQual + ns(GarageCars) + BsmtQual +
##     MSSubClass + ns(age)
##
##                    Df Sum of Sq  RSS   AIC
## <none>                          35.3 -5390
## + ExterQual         1      0.03 35.2 -5389
## - BsmtQual          1      0.08 35.3 -5389
## + MoSold           11      0.49 34.8 -5389
## - KitchenQual       1      0.43 35.7 -5375
## - ns(age)           1      0.69 36.0 -5364
## - Neighclass        1      0.71 36.0 -5363
## - ns(TotBathrooms)  1      2.45 37.7 -5294
## - ns(GarageCars)    1      2.50 37.8 -5292
## - MSSubClass       14      4.92 40.2 -5227
## - OverallQual       1      5.86 41.1 -5168
## - ns(TotalArea)     1      6.41 41.7 -5148
```

7.4 Categorical regression model

Since there are categorical variables in model, I wii try categorical regression model. Categorical regression quantifies categorical data by assigning numerical values to the categories, resulting in an optimal linear regression equation for the transformed variables. Categorical regression is also known by the acronym CATREG, for *cat*egorical *reg*ression. In categorical model the degree parameter specifies the polynomial degree of the B-spline basis for each dimension of the continuous x (default degree=3, i.e. cubic spline). So here I choose the defult one.

```
cat_reg_splines<- crs(SalePrice ~ TotalArea + OverallQual+ Neighclass+ TotBat
hrooms+ KitchenQual+ GarageCars+ BsmtQual+ ExterQual+ MSSubClass+ MoSold+ ag
e,
              data=table,#integer/vector specifying the polynomial degree of
the B-spline basis for each dimension of the continuous x (default degree=3,
i.e. cubic spline),
              cv="none",
              kernel=TRUE)

summary(cat_reg_splines)
```

```
## Call:
## crs.formula(formula = SalePrice ~ TotalArea + OverallQual + Neighclass +
##     TotBathrooms + KitchenQual + GarageCars + BsmtQual + ExterQual +
##     MSSubClass + MoSold + age, data = table, cv = "none", kernel = TRUE)
##
## Residual standard error: 0.05075 on 1459 degrees of freedom
## Multiple R-squared: 0.9839,   Adjusted R-squared: 0.9839
## F-statistic: NA on NA and NA DF, p-value: NA
## Estimation time: 4.3 seconds
```

The R-squared of crs is 0.9839, which is a good result.

7.5 Lasso with cross validation

Before fitting lasso model, I use one-hot encoding for categorical variables. I use cross validation to choose value of alpha and lambda. It turns out that the best value for alpha is 1 and for lambda is 0.05. The RMSE of Lasso model is 0.141.

```
set.seed(27042018)
my_control <-trainControl(method="cv", number=5)
lassoGrid <- expand.grid(alpha = 1, lambda = seq(0.001,0.1,by = 0.0005))

lasso_mod <- train(x=com_var2, y=table$SalePrice[!is.na(table$SalePrice)], me
thod='glmnet', trControl= my_control, tuneGrid=lassoGrid)
lasso_mod$bestTune
```

```
##   alpha lambda
## 9     1  0.005
```

```
min(lasso_mod$results$RMSE)

## [1] 0.141
```

The most important 20 variables given by Lasso model is present below, which is not consistent with results by random forest.

```
varImp(lasso_mod)

## glmnet variable importance
##
##   only 20 most important variables shown (out of 145)
##
##                          Overall
## MSSubClass1 story 1945-   100.0
## OverallQual                91.1
## CentralAir                 84.6
## Condition2Norm             82.6
## Exterior1stBrkFace         82.4
## StreetPave                 82.1
## SaleTypeNew                74.7
## MSZoningRL                 74.7
## GarageCars                 59.9
## MSZoningFV                 58.0
## TotBathrooms               56.2
## Condition1Norm             54.1
## LotArea                    49.0
## OverallCond                40.2
## SaleConditionNormal        40.1
## TotRmsAbvGrd               37.6
## MSZoningRH                 34.0
## FoundationPConc            33.5
## PavedDrive                 33.1
## LotConfigCulDSac           31.3
```

7.6 Gradient Boosting

```
library(gbm)

gbm_model <- gbm(SalePrice ~., data=table, distribution=="laplace", n.tree=10
0)
pred_gbm <- predict(gbm_model, table, n.tree=100)

RMSE3 <- RMSE(predict_gbm, table$SalePrice)

[1] 0.148
```

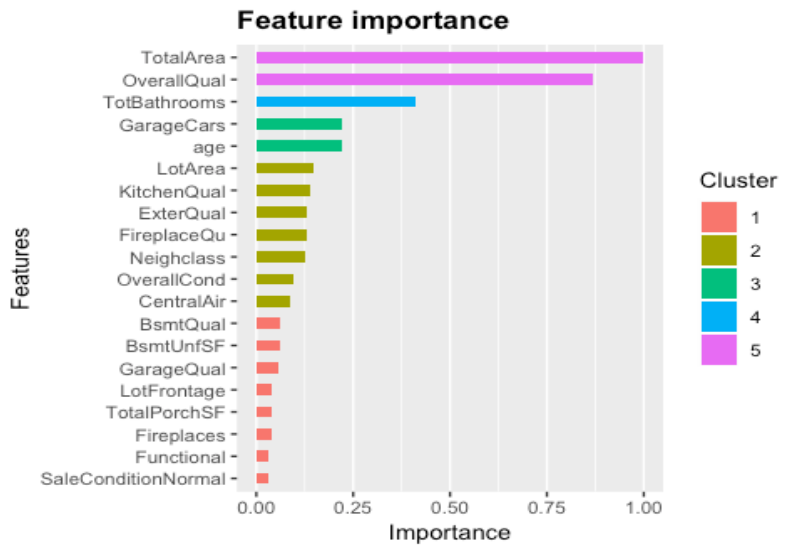
7.7 XGboost with cross validation

I first use cross validation to choose eta, max_depth, min_child_weight and the values of parameters are given below. Then I train model with XGboost method. And at last the RMSE of XGboost is 0.128602. And the feature importance given by XGboost is almost consistent with that given by Random Forest.

```
##    nrounds max_depth eta gamma colsample_bytree min_child_weight subsample
## 31    1000         5 0.1     0                1                3         1
```

```
df_xgboost <- xgb.DMatrix(data = as.matrix(com_var2), label= table$SalePrice)
xgbcv <- xgb.cv( params = xgb_params, data =df_xgboost, nrounds = 500, nfold
= 5, showsd = T, stratified = T, print_every_n = 40, early_stopping_rounds =
10, maximize = F)
```

```
## [1]  train-rmse:10.955492+0.004347   test-rmse:10.955480+0.017843
## Multiple eval metrics are present. Will use test_rmse for early stopping.
## Will train until test_rmse hasn't improved in 10 rounds.
##
## [41] train-rmse:1.427145+0.000529    test-rmse:1.428089+0.014497
## [81] train-rmse:0.222758+0.000995    test-rmse:0.234616+0.011947
## [121]    train-rmse:0.104141+0.001710    test-rmse:0.136344+0.011380
## [161]    train-rmse:0.090226+0.001420    test-rmse:0.131140+0.012598
## [201]    train-rmse:0.082927+0.001454    test-rmse:0.129741+0.013486
## [241]    train-rmse:0.077013+0.001602    test-rmse:0.129226+0.013877
## [281]    train-rmse:0.072220+0.001514    test-rmse:0.128888+0.014257
## [321]    train-rmse:0.068269+0.001536    test-rmse:0.128661+0.014293
## Stopping. Best iteration:
## [315]    train-rmse:0.068812+0.001518    test-rmse:0.128602+0.014279
```



8. Conlusion

	Random Forest	GAM	Natural Cubic Spline	CRS	Lasso	Gradient boosting	XGboost
R square	0.867	0.847	0.847	0.984			
RMSE	0.157				0.141	0.148	0.129

It can be concluded that in tree methods XGboost performs the best. And feature importance give by XGboost and Random Forest are almost consistent with each other. For smooth methods, the natural cubic spline and CRS are the best choices. Among all methods, the Lasso gives the smallest RMSE value (0.141). However, the feature importance calculated by Lasso is much different from that by Random Forest and XGboost.