**The Problem**

1. 问题描述

通过房屋的一些属性来预测房价

1. 分析目的

通过构建回归模型来预测房价，并通过研究房屋的哪些属性对房价的影响较大，来影响房地产投资决策

1. 数据集

3.1数据集简介

我们选择使用Ames Housing数据集来构建我们的模型。Ames Housing数据集由De Cock(Truman State University 统计学教授)收集，包含了79个解释变量，描述了爱荷华州埃姆斯(Ames, Iowa)2919处住宅的方方面面。各个变量的详细信息请查看data\_description.txt.

3.2 数据集来源

本数据来源于数据科学网站Kaggle(www.kaggle.com)

3.3 可靠性

本数据集已经是机器学习和数据分析领域得到广泛应用的数据集，得到了诸多数据科学家和数据科学竞赛的认可。

3.4 时效性

本数据集没有提到记录时间，但有时间相关的属性

1. 回归分析

我们通过解释变量来对房价进行回归

1. 目标变量

我们选择目标变量是房价，因为我们认为房价是房地产投资中最重要的指标。

流程

建模流程主要包含如下7个内容：

1. 载入数据和简单探索
2. 挖掘重要变量(Understand the data)
3. 数据处理与编码(Understand the data,Data Preparation)
4. 重要变量可视化(Understand the data)
5. 特征工程(Data Preparation)
6. 数据预处理(Data Preparation)
7. 建模与评估(Prediction Models)

1 载入数据并进行简单探索

1.1 载入必要的R包

1. library(knitr)
2. library(ggplot2)
3. library(plyr)
4. library(dplyr)
5. library(corrplot)
6. library(caret)
7. library(gridExtra)
8. library(scales)
9. library(Rmisc)
10. library(ggrepel)
11. library(randomForest)
12. library(psych)
13. library(xgboost)

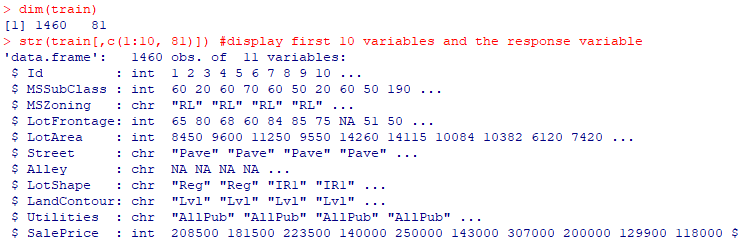
1.2 读取数据

将CSV文件读取为Dataframe

1. data <- read.csv("C:/Users/asus/Desktop/HPART/train.csv",stringsAsFactors=F)
2. train <- data[1:1022,]
3. test <- data[1023:1460,]
4. test\_label = test$SalePrice
5. test$SalePrice <- 'None'

1.3 数据集大小与结构

1. dim(train)
2. str(train[,c(1:10, 81)]) #显示前10个属性和预测数据(房价)



训练集共1460条数据, 80个属性.

1.4 合并

1. test\_labels <- test$Id
2. test$Id <- NULL
3. train$Id <- NULL
4. test$SalePrice <- NA
5. all <- rbind(train, test)
6. dim(all)

合并数据集得到了2919条数据,80个属性.

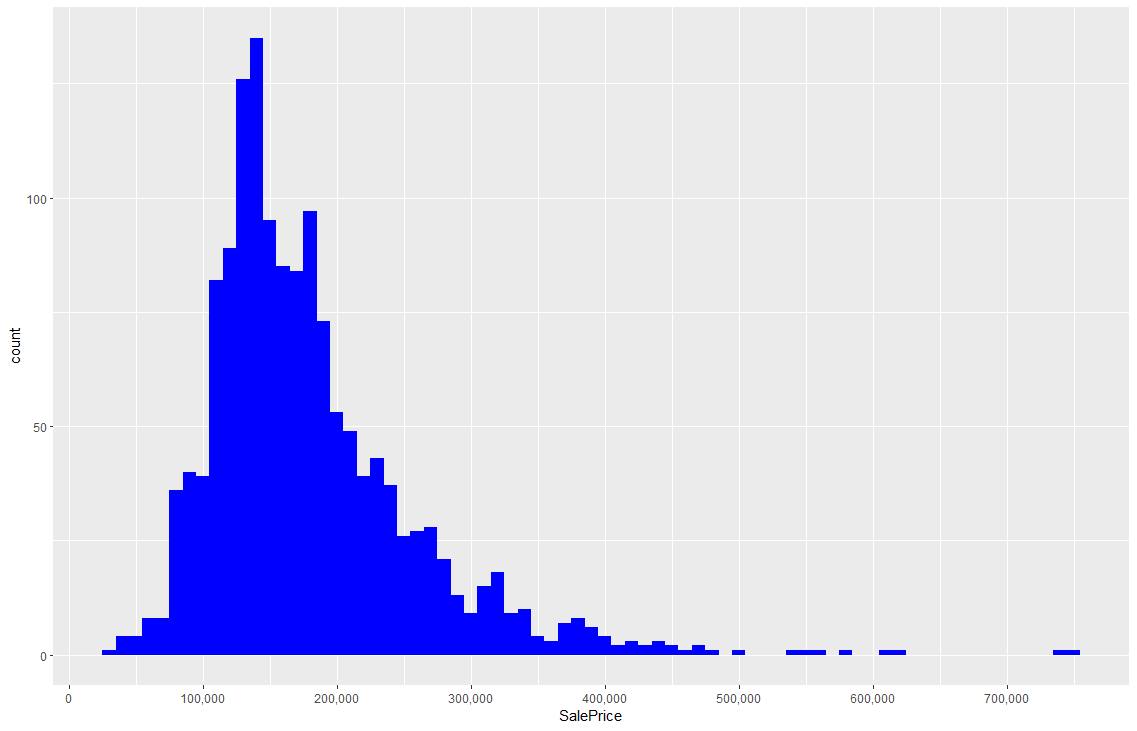


2 挖掘重要变量

2.1目标变量的分布

绘制房价的频数分布直方图, 这也符合少数人买得起高价房的经验.在建模时可能会进行处理.

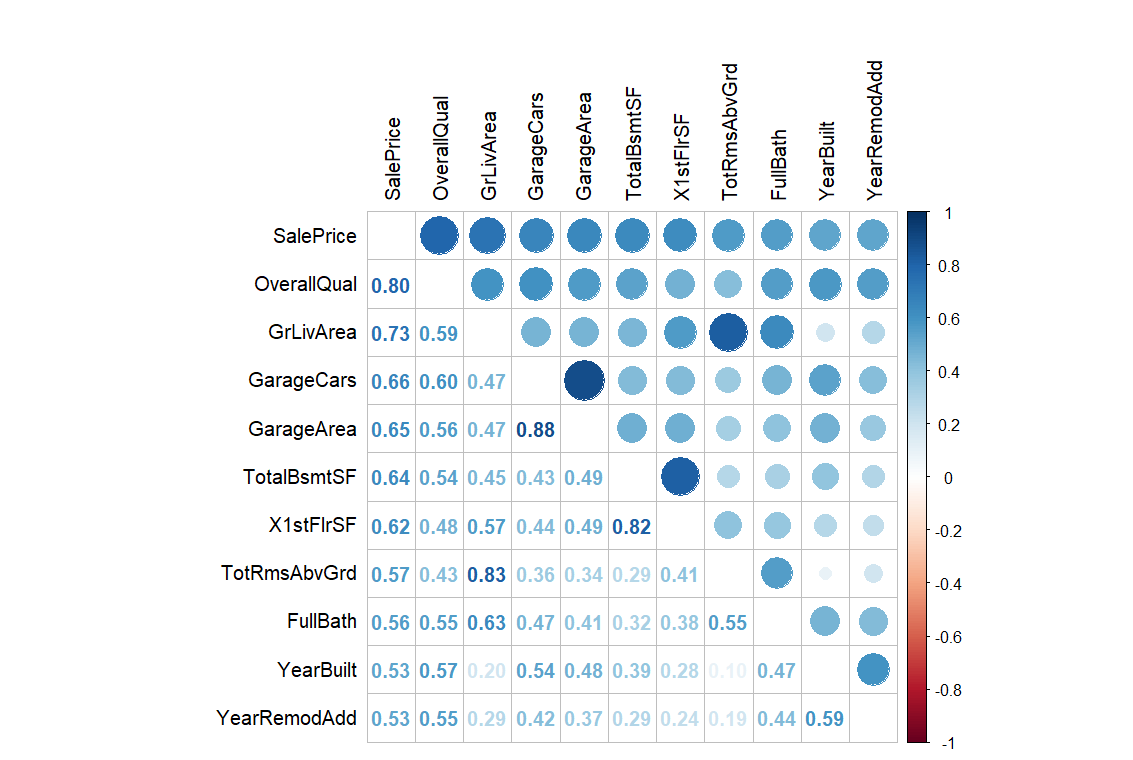
1. ggplot(data=all[!is.na(all$SalePrice),], aes(x=SalePrice)) +
2. geom\_histogram(fill="blue", binwidth = 10000) +
3. scale\_x\_continuous(breaks= seq(0, 800000, by=100000), labels = comma)
4. summary(all$SalePrice)



2.2 重要相关数字型变量

因为原始数据没有经过预处理,很多变量不是数字型的,为了先对数据集有个大致的认识,先求数字型变量的相关矩阵.

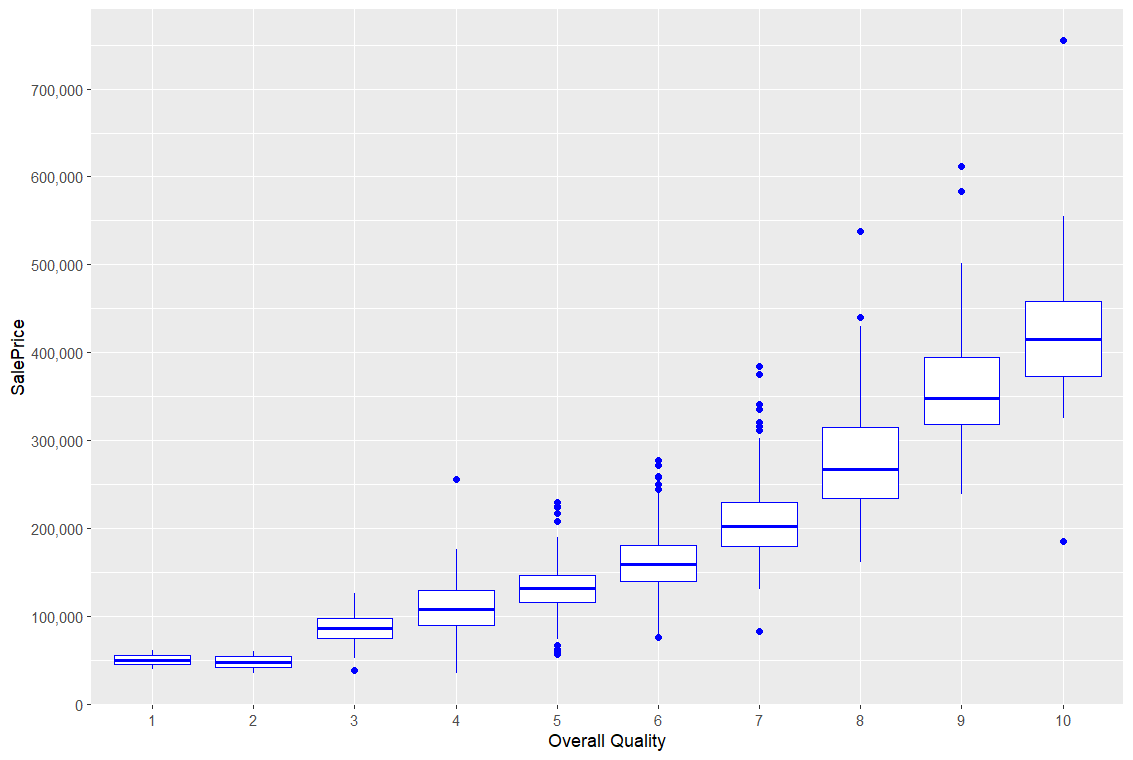
1. numericVars <- which(sapply(all, is.numeric)) *#求出数字型变量的下标*
2. numericVarNames <- names(numericVars) *#存储数字型变量的变量名*
3. cat('There are', length(numericVars), 'numeric variables')
4. all\_numVar <- all[, numericVars]
5. cor\_numVar <- cor(all\_numVar, use="pairwise.complete.obs") *#相关性*
6. *#根据与房价的相关性大小排序*
7. cor\_sorted <- as.matrix(sort(cor\_numVar[,'SalePrice'], decreasing = TRUE))
8. *#选出相关性大于0.5的变量*
9. CorHigh <- names(which(apply(cor\_sorted, 1, function(x) abs(x)>0.5)))
10. cor\_numVar <- cor\_numVar[CorHigh, CorHigh]
11. corrplot.mixed(cor\_numVar, tl.col="black", tl.pos = "lt")



2.3 Overall Quality

Overall Quality与房价相关性最大,先对这个变量进行分析

1. ggplot(data=all[!is.na(all$SalePrice),], aes(x=factor(OverallQual), y=SalePrice))+
2. geom\_boxplot(col='blue') + labs(x='Overall Quality') +
3. scale\_y\_continuous(breaks= seq(0, 800000, by=100000), labels = comma)



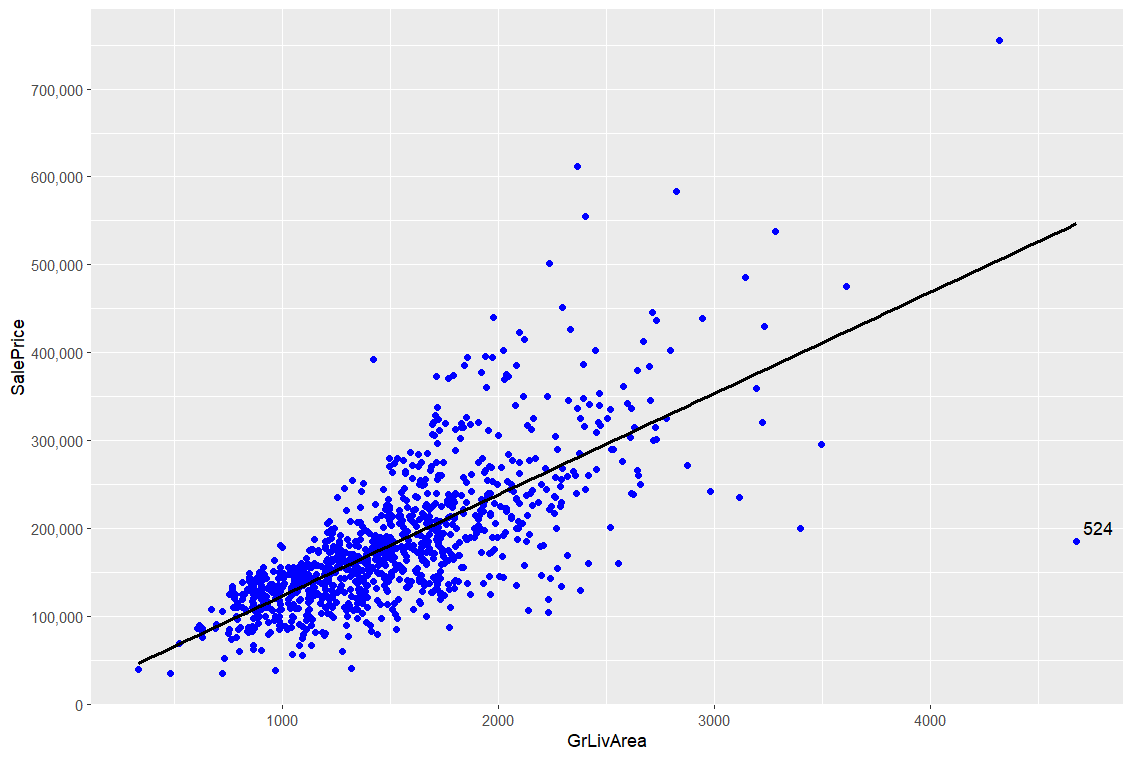
在数值变量中，房屋的整体质量与售价之间的相关性最高（0.79）。它根据一个从1（非常差）到10（非常优秀）的评分标准，对房屋的整体材料和装修进行评级。这种正相关确实存在，并且似乎是一个略微上升的曲线。

2.4 Above Grade (Ground) Living Area (square feet)

绘制GrLvArea与房价的散点图

1. ggplot(data=all[!is.na(all$SalePrice),], aes(x=GrLivArea, y=SalePrice))+
2. geom\_point(col='blue') + geom\_smooth(method = "lm", se=FALSE, color="black", aes(group=1)) +
3. scale\_y\_continuous(breaks= seq(0, 800000, by=100000), labels = comma) +
4. geom\_text\_repel(aes(label = ifelse(all$GrLivArea[!is.na(all$SalePrice)]>4500, rownames(all), '')))
5. all[c(524, 1299), c('SalePrice', 'GrLivArea', 'OverallQual')]

相关性的第二高的是Above Grade Living Area, 实际上越大的房越贵也符合生活经验.



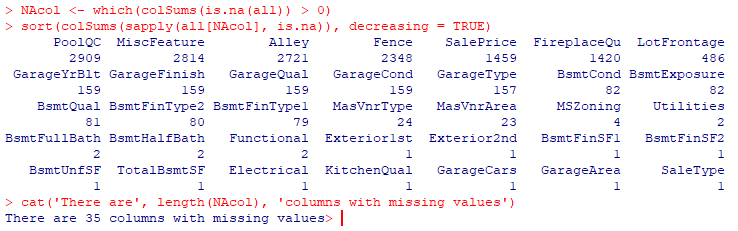
图中有一些售价便宜面积大的离群点(524和1299),我们对其做进一步的观察



这两条数据比较反常,因为其房子总体质量也很好.在之后可能会将其清除.

3 数据处理与编码

3.1 缺失值检查



共20个属性含有缺失值，因此接下来需要处理其中19个属性中的缺失值(还有一个是目标变量)

3.2缺失值与编码

本节将对缺失值进行处理，因为涉及到的变量过多，我在Excel中简要介绍了处理方式，详细处理方式请检查脚本中的代码。

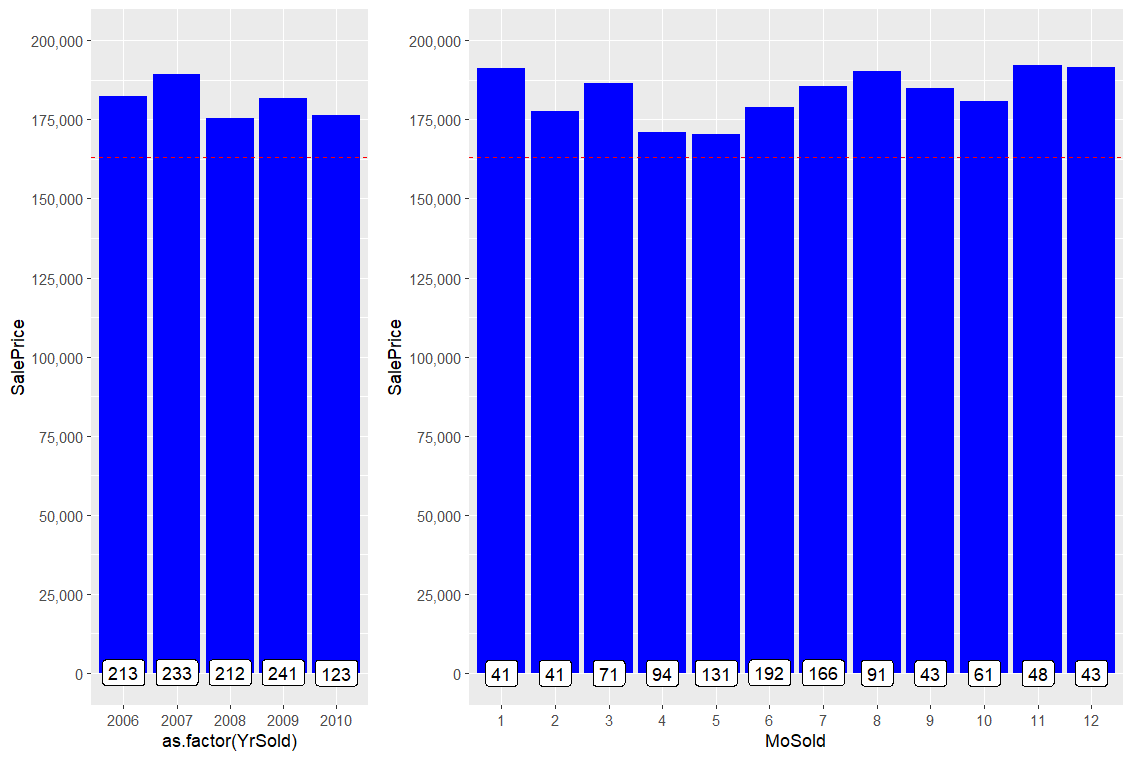
3.3 部分数值型变量转换

当前我们已经填充了所有的缺失值，并完成了大多数编码工作，但仍有三个数值型变量应该被编码为类别类型。

3.3.1 销售时间(Year and Month Sold)

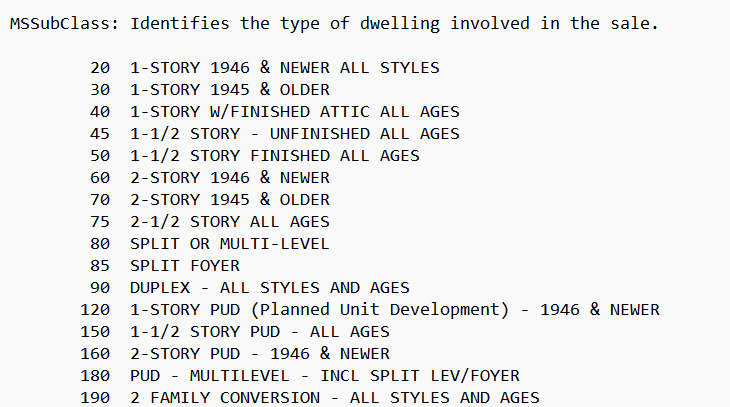
虽然根据老房子价值较低，新房价值相对高这一社会经验进行编码有其合理之处，但我们没考虑到一些其他的因素。例如：2009年的销售价格很可能比2007年(2007年发生了经济危机)低得多。我们将在建模之前将YrSold转换为因子，但由于我们还需要数字版本的YrSold来创建Age变量，因此我还没有这样做。Month Sold也是一个整数变量。但是，12月并不比1月好。因此，我将MoSold值转换回因子。

1. str(all$YrSold)
2. str(all$MoSold)
3. all$MoSold <- as.factor(all$MoSold)
4. ys <- ggplot(all[!is.na(all$SalePrice),], aes(x=as.factor(YrSold), y=SalePrice)) +
5. geom\_bar(stat='summary', fun.y = "median", fill='blue')+
6. scale\_y\_continuous(breaks= seq(0, 800000, by=25000), labels = comma) +
7. geom\_label(stat = "count", aes(label = ..count.., y = ..count..)) +
8. coord\_cartesian(ylim = c(0, 200000)) +
9. geom\_hline(yintercept=163000, linetype="dashed", color = "red") *#dashed line is median SalePrice*
10. ms <- ggplot(all[!is.na(all$SalePrice),], aes(x=MoSold, y=SalePrice)) +
11. geom\_bar(stat='summary', fun.y = "median", fill='blue')+
12. scale\_y\_continuous(breaks= seq(0, 800000, by=25000), labels = comma) +
13. geom\_label(stat = "count", aes(label = ..count.., y = ..count..)) +
14. coord\_cartesian(ylim = c(0, 200000)) +
15. geom\_hline(yintercept=163000, linetype="dashed", color = "red") *#dashed line is median SalePrice*
16. grid.arrange(ys, ms, widths=c(1,2))



### 3.3.2 MSSubClass

1. all$MSSubClass <- as.factor(all$MSSubClass)
2. *#revalue for better readability*
3. all$MSSubClass<-revalue(all$MSSubClass, c('20'='1 story 1946+', '30'='1 story 1945-', '40'='1 story unf attic', '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'='split 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'))
4. str(all$MSSubClass)



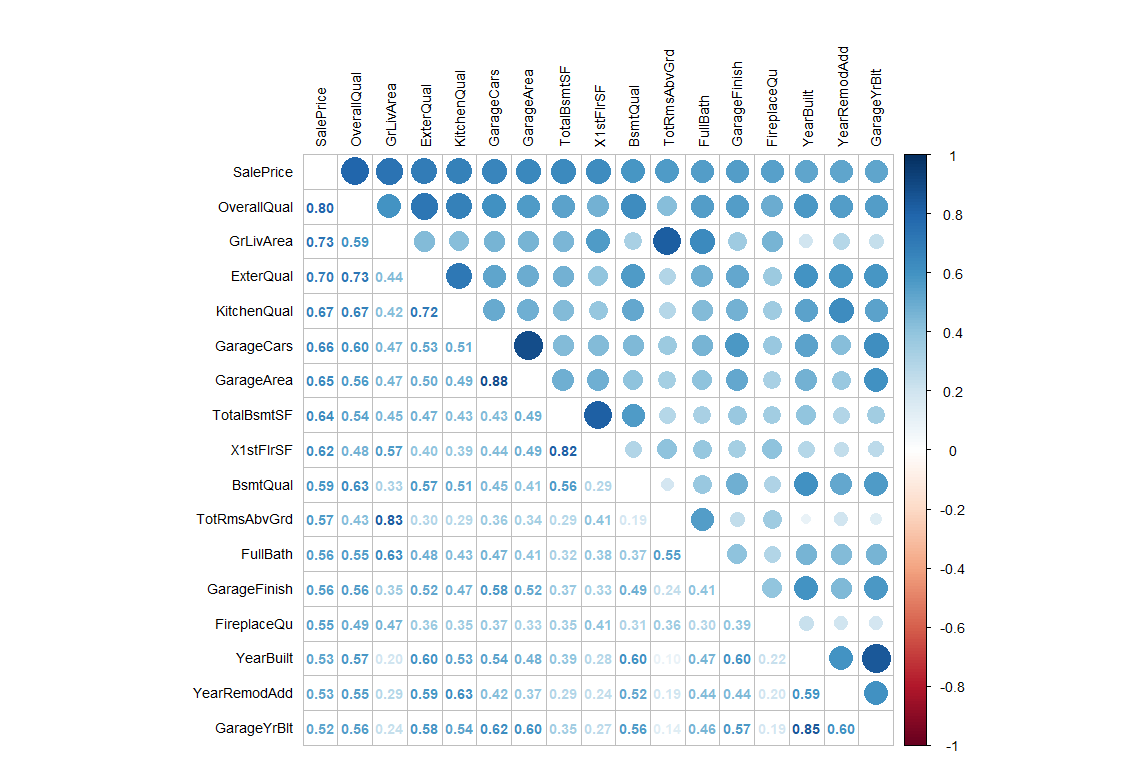
这个变量实际上需要编码成类别的类型

4 重要变量可视化

4.1 重新绘制相关矩阵

前一节我们将所有字符串类型转为了类别类型或数字类型，并完成部分数字型变量编码为类别变量的工作，并清除了Utilities变量。现在我们有56个数字变量和23个类别变量。

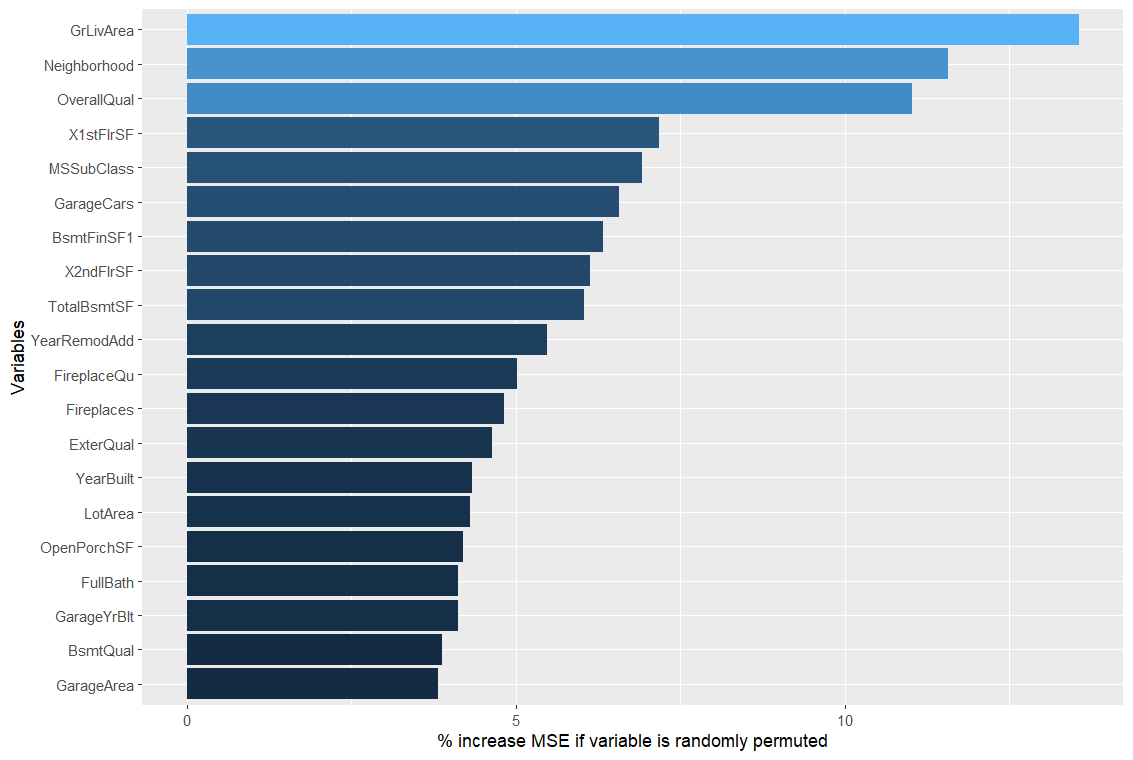
我们重新求一次数字型变量的相关矩阵，绘制出相关性大于0.5的变量



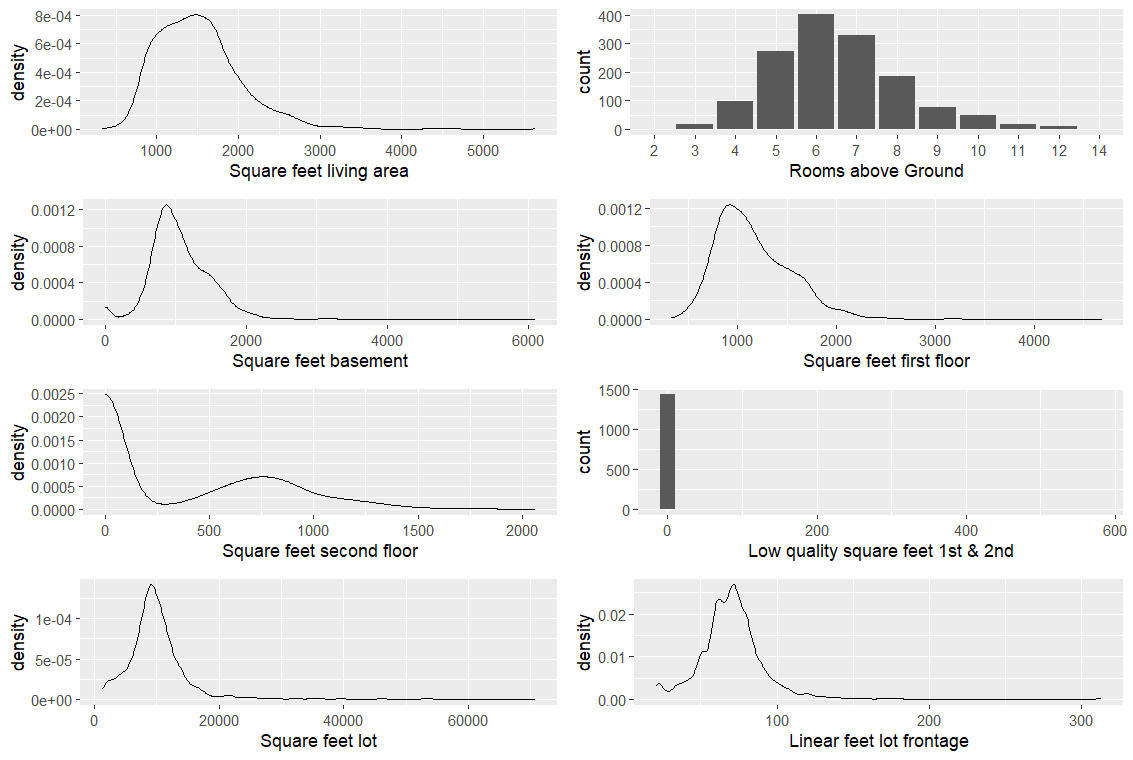
4.2 通过随机森林算法快速搜索重要性最高的变量

尽管相关矩阵可以很好的描述数字型变量之间的共线性关系，但不能描述分类变量与售价的关系。我们决定通过随机森林来获取变量重要性。

下图中仅有Neighborhood, MSSubClass是类别变量.



4.2.1 面积相关的变量的分布



我们再研究GrLivArea，X1stFlrSF，X2ndFlrSF，LowQualFinSF四个变量之间的关系

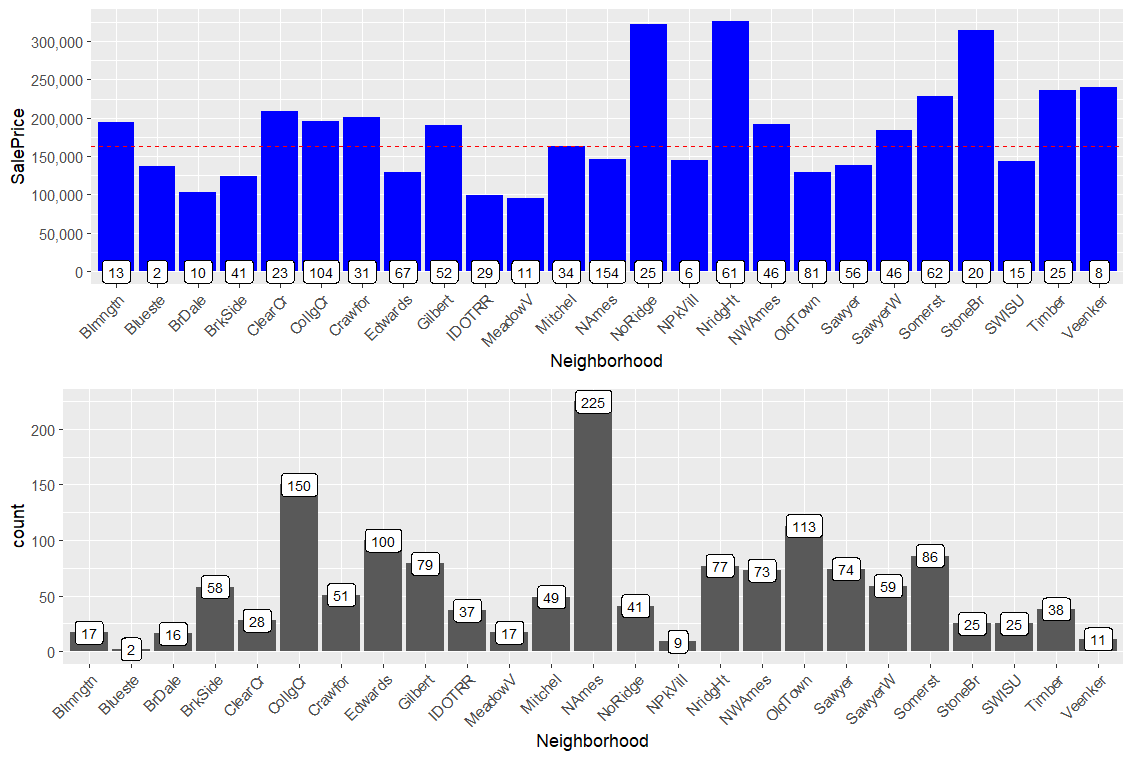
1. cor(all$GrLivArea, (all$X1stFlrSF + all$X2ndFlrSF + all$LowQualFinSF))

事实证明，这三个面积变量包含在了GrLivArea中，因为这三个变量与GrLivArea之间的相关性为1

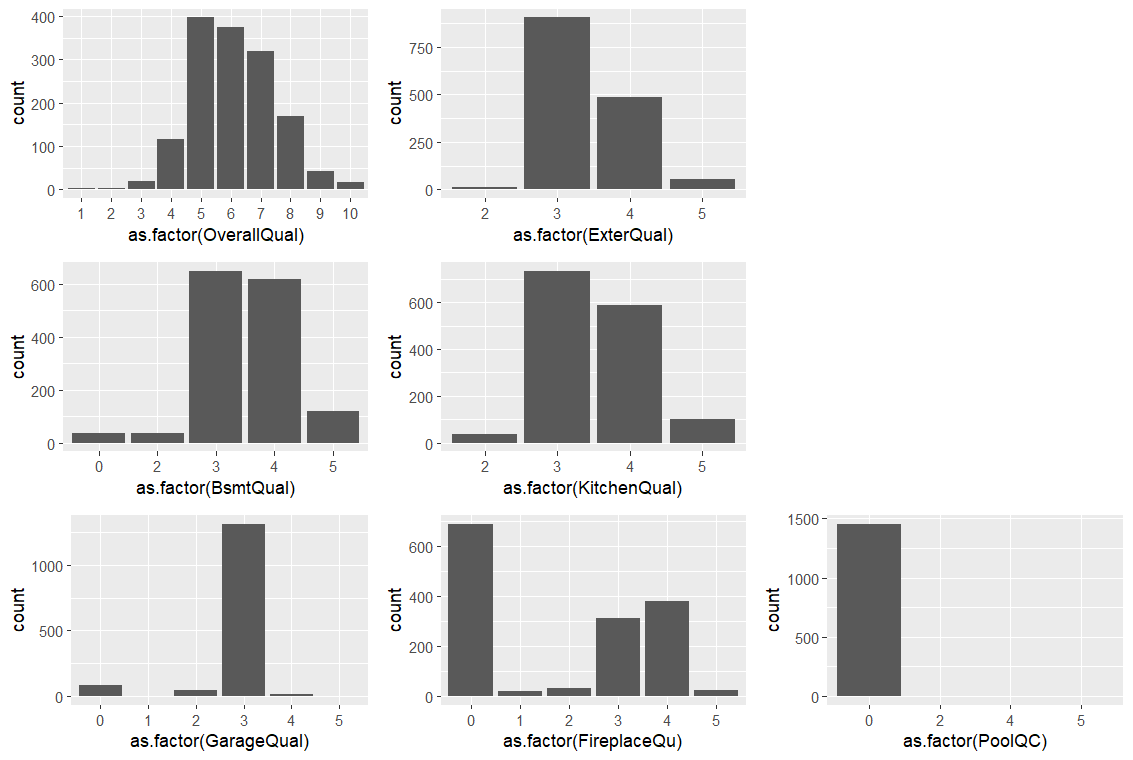


4.2.2 探索最重要的类别变量Neighborhood

上面的图展示了各类别中售价的中位数。下面的图展示了各类别的频数。



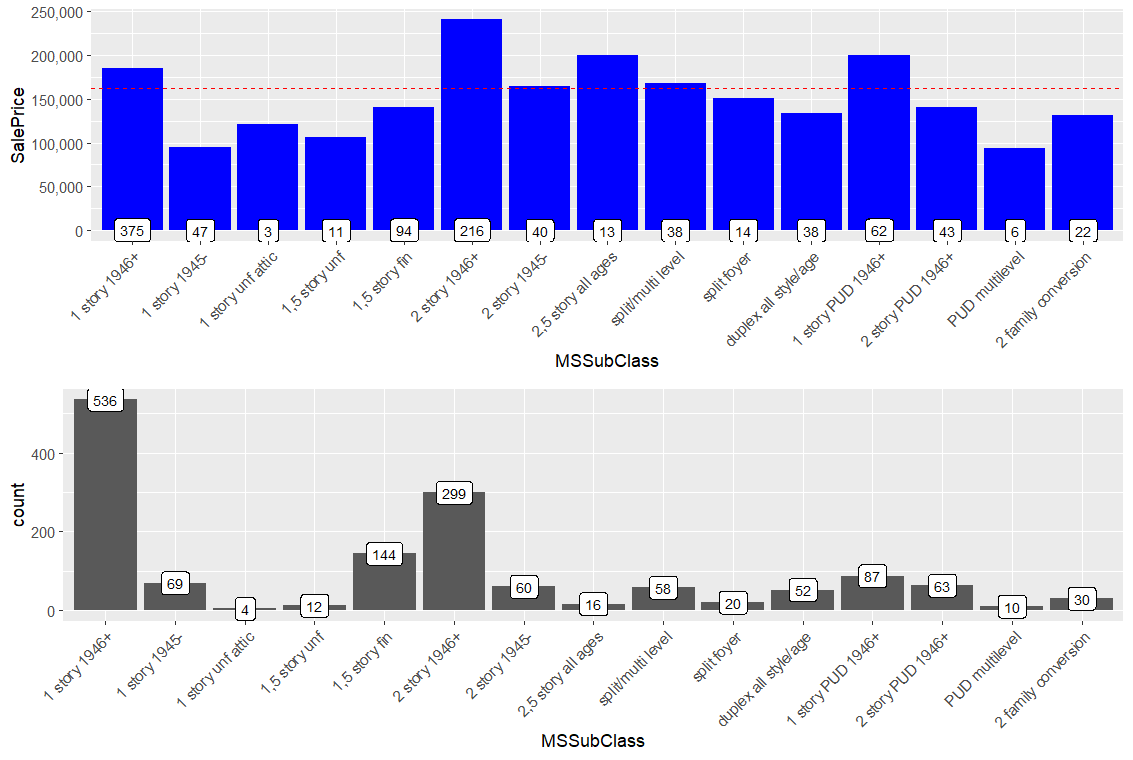
4.2.3 探索总质量与其他质量变量



Overall Quality是一个相当重要的变量，ExterQual与Overall Quality的相关性很高，厨房质量似乎也值得保留，车库似乎区别不大，而壁炉质量的相关性较高。PoolQC过于稀疏，可能会重新增加一个has pool的变量。

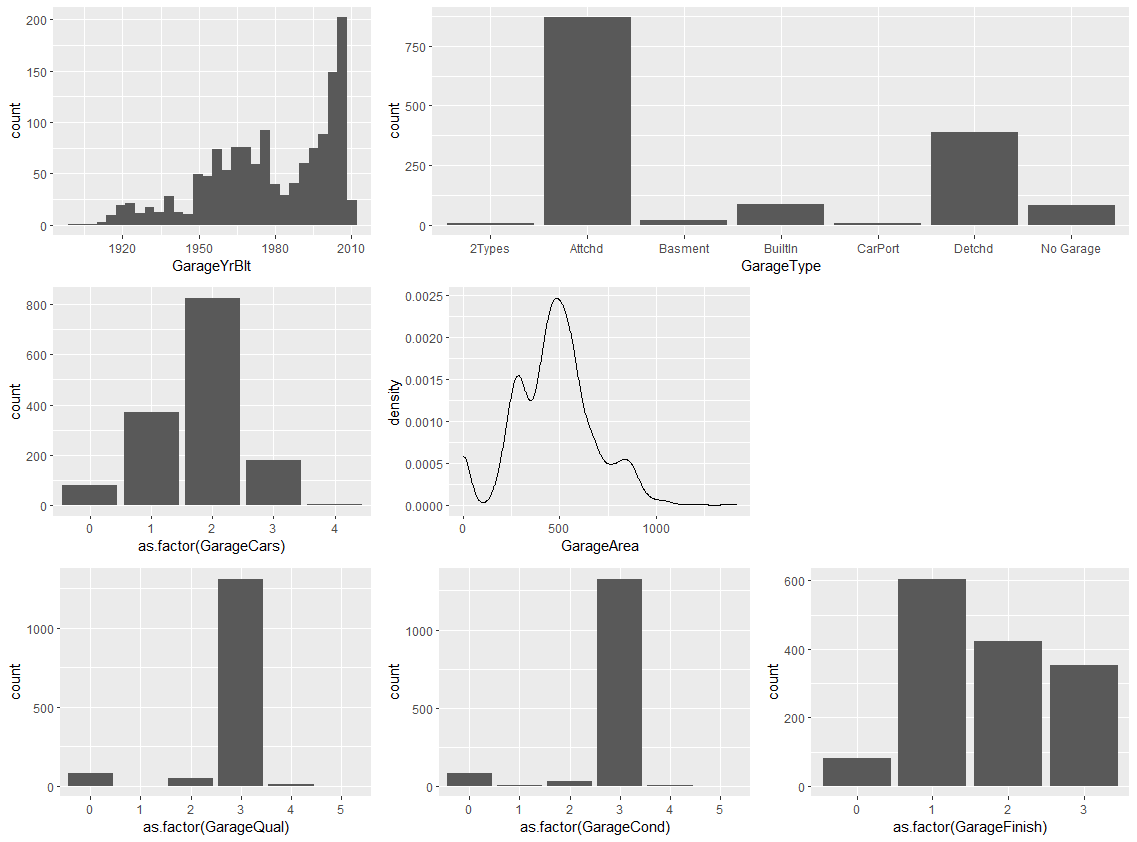
### 4.2.4 第二重要的类别变量MSSubClass

此处图表的含义与6.2.2相同



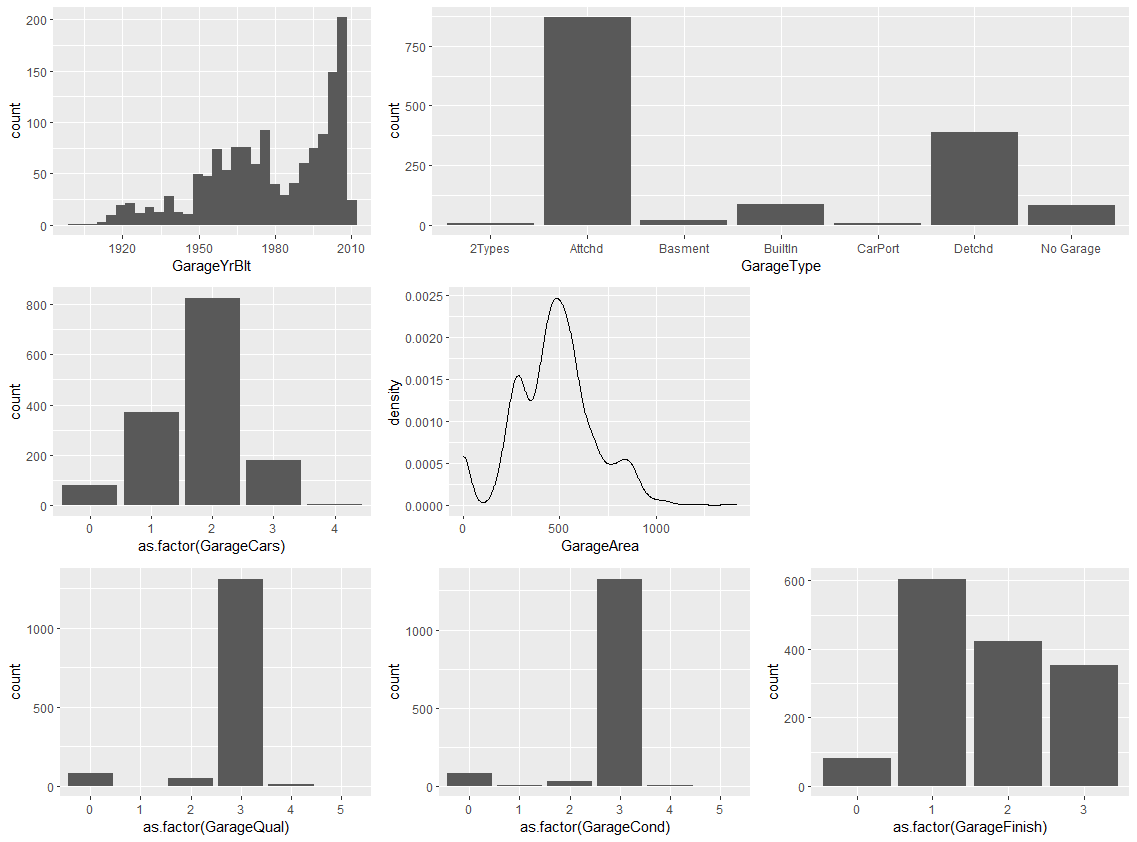
4.2.5 探索车库相关变量

观察下图，发现GarageCars和GarageArea似乎有较强的相关性(符合常理), 而GarageQual和GarageCond也有比较强的相关性。



4.2.6 探索地基相关变量

总共有11个相关变量，我们挑选其中的8个进行观察。



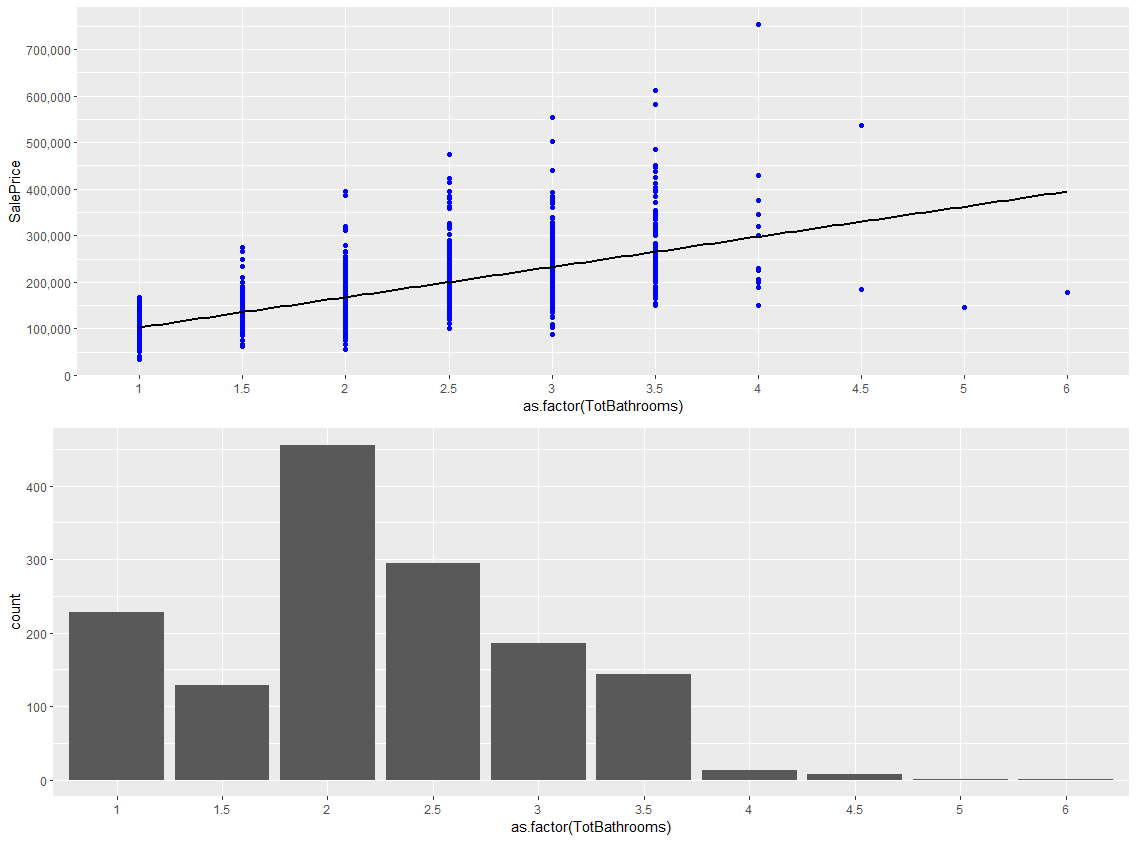
看起来平方英尺的总地下室表面（TotalBsmtSF）进一步分解为完成区域（如果有多种类型的完成，则为2个）和未完成区域。我检查了这3个变量的总和与TotalBsmtSF之间的相关性。相关性恰好为1。

5 特征工程

5.1 卧室数量

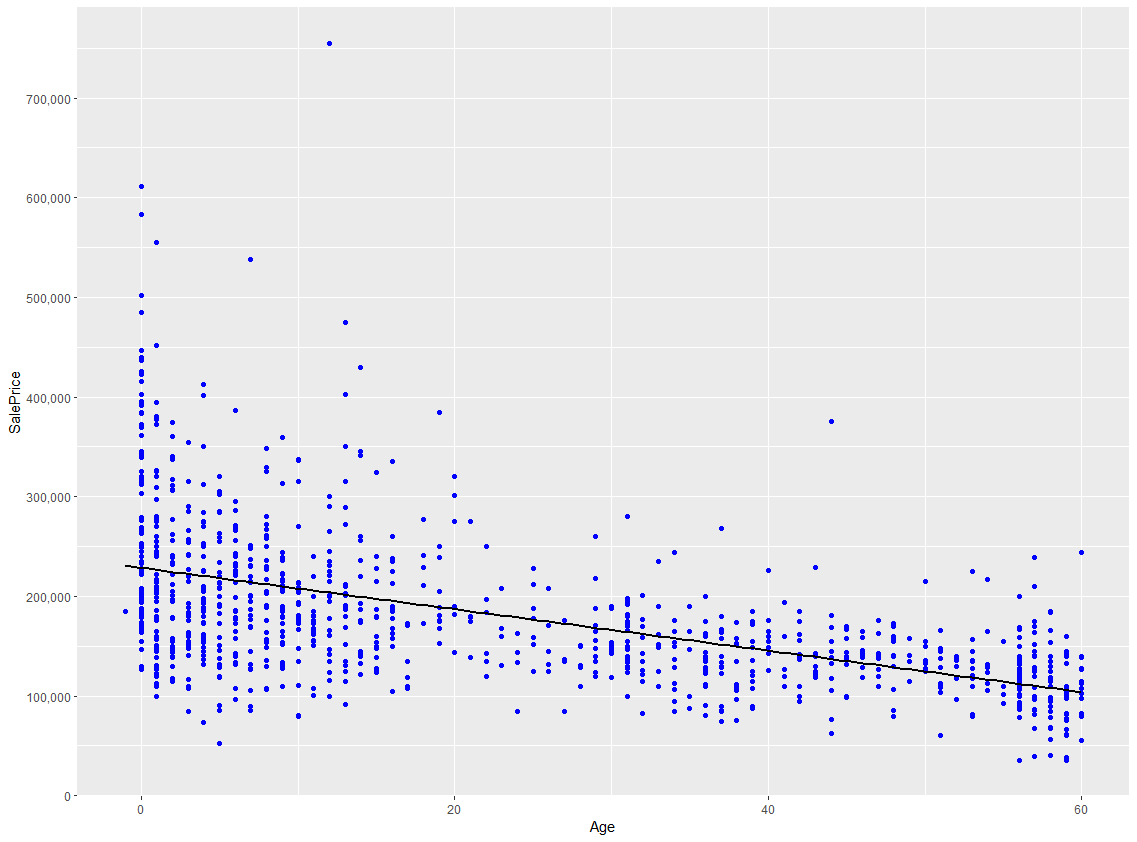
与卧室相关的变量有4个，都不重要(在前面的相关矩阵中找不到)，我们推测按照如下规则将这些变量加起来之后的新变量TotalBathRooms可能是一个相关性强的变量。

观察下表，可以发现似乎有了一些相关性。

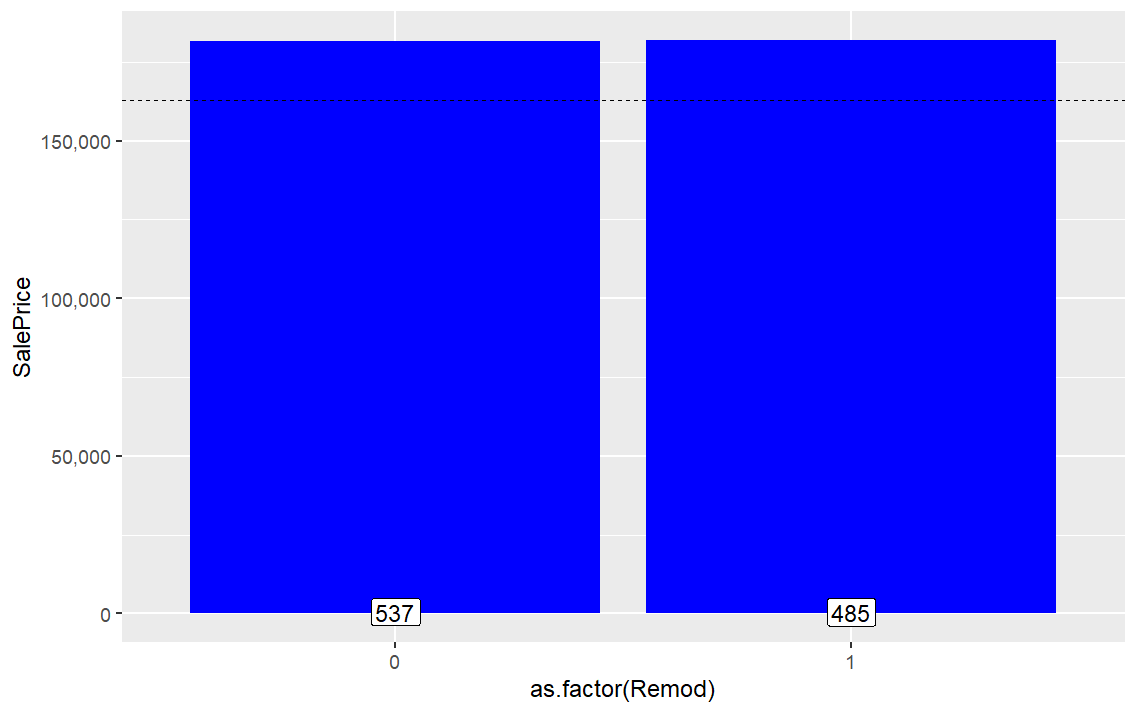


5.2 添加House Age, Remodeled，IsNew变量

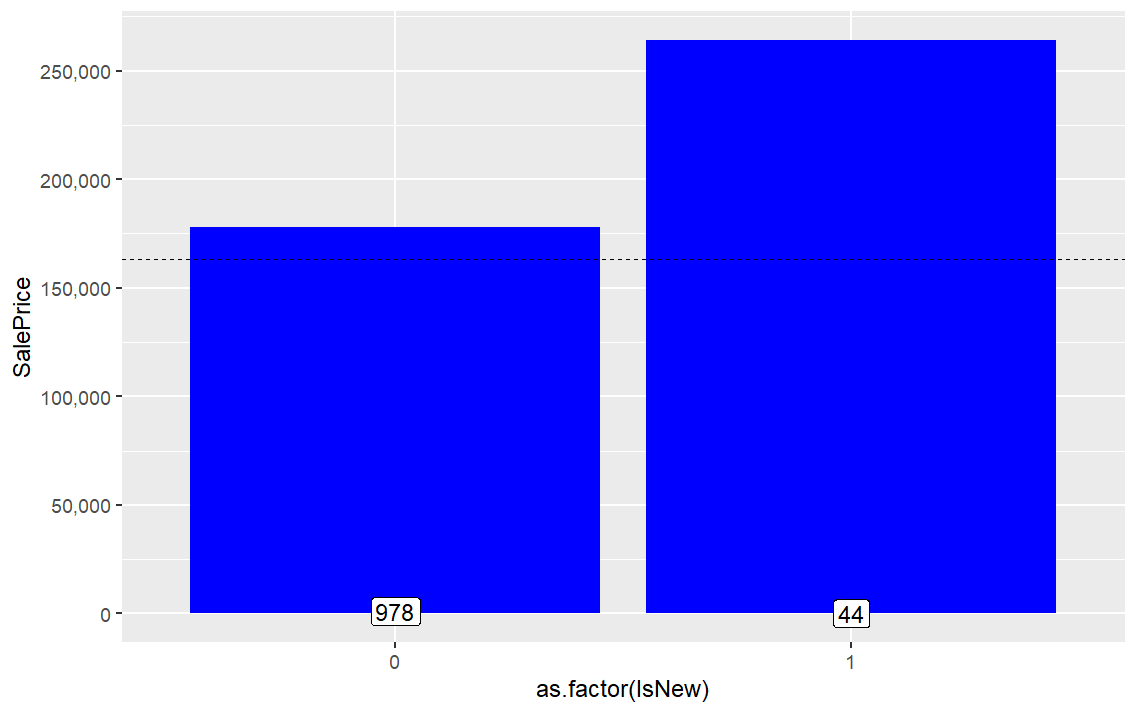
总的来说，有3个变量与房屋的年龄有关；YearBlt，YearRemodAdd和YearSold。如果没有翻新/添加，则YearRemodAdd默认为YearBuilt。我们使用YearRemodeled和YearSold来确定年龄。然而，由于旧建筑的部分始终会保留，而只有房屋的部分可能已经翻新，因此我还将引入一个Remodeled Yes / No变量。这应该被视为某种惩罚参数，表明如果年龄是基于翻新日期的，则可能不如在同一年建造的从头开始建造的房屋有价值。



房龄与价格呈现出负相关。



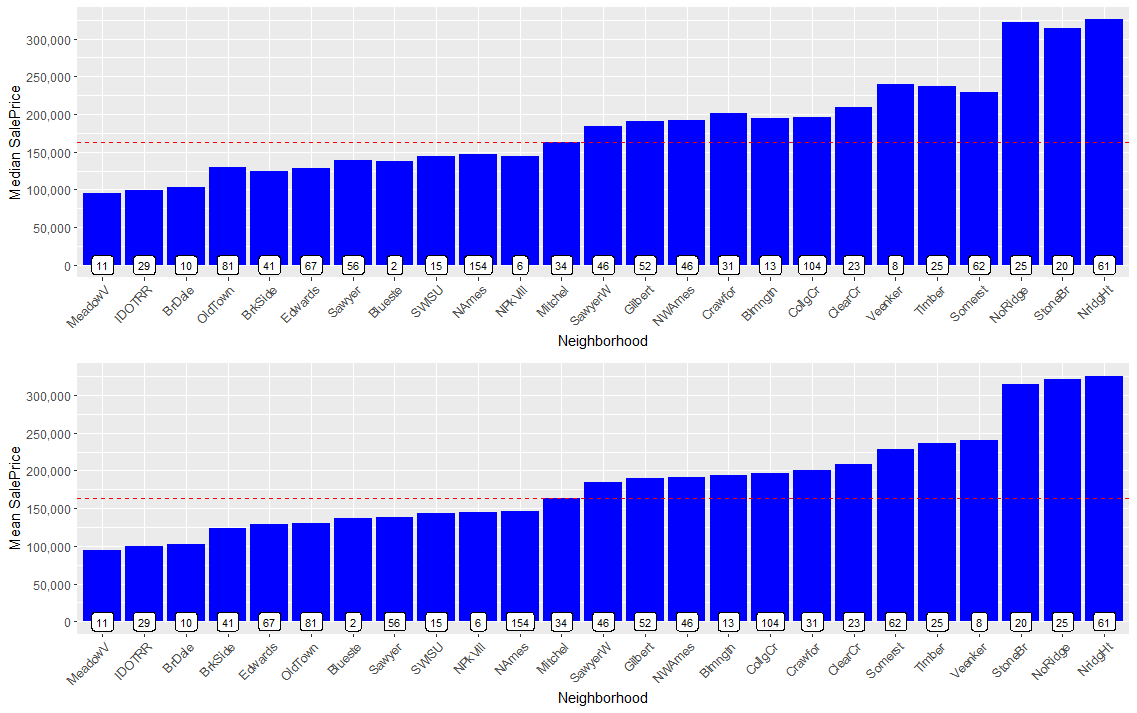
翻修过后房价没有明显变化。



新房房价更高。

5.3 对Neighborhood分组

将Neighborhood从高到低分为三组：最好的三个分为2，最差的三个分为0，其余的分为1.



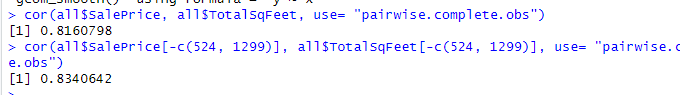
## 5.4 Total Square Feet

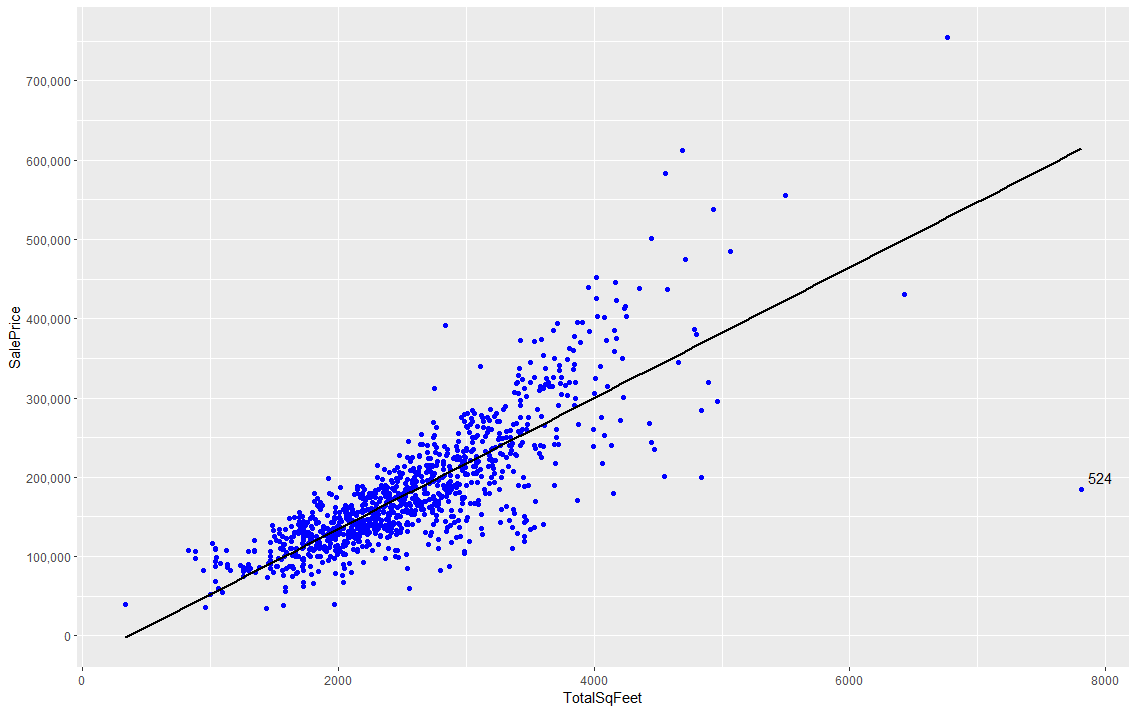
我们加入新变量Total Square Feet

检验一下相关性：

1. cor(all$SalePrice, all$TotalSqFeet, use= "pairwise.complete.obs")
2. cor(all$SalePrice[-c(524, 1299)], all$TotalSqFeet[-c(524, 1299)], use= "pairwise.complete.obs")

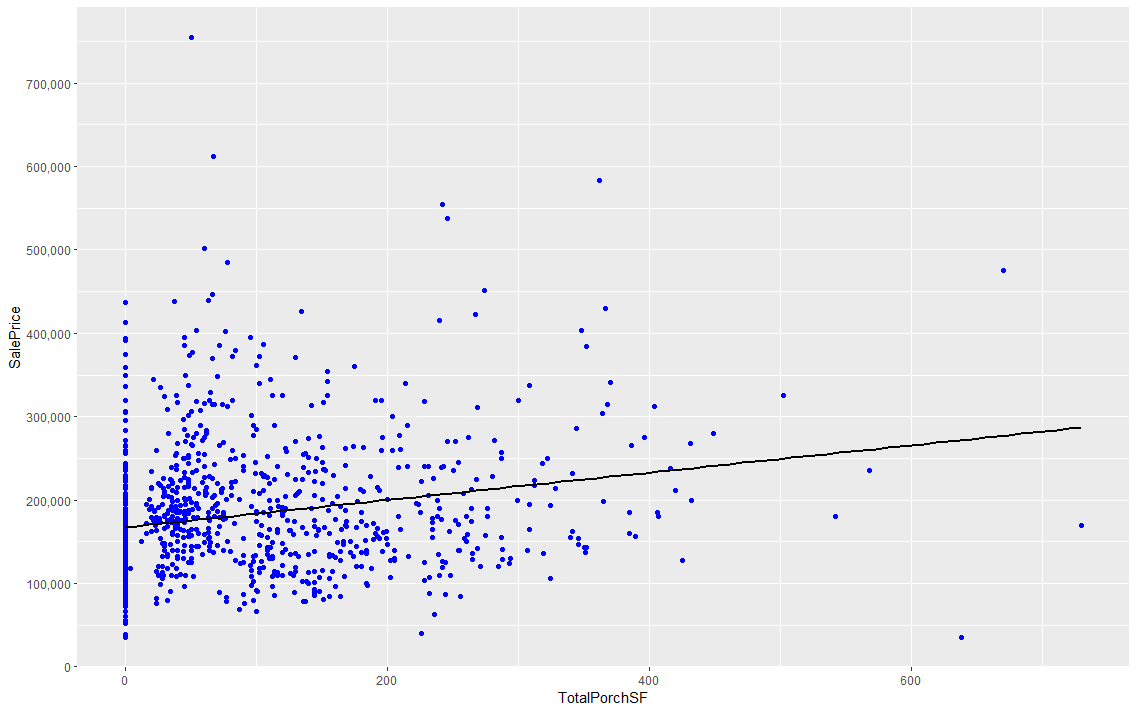
清除离群点后得到的相关性提升了2%。





## 5.5 合并门廊变量

加入TotalPorchSF变量



与房价相关性为21%。

6 数据准备

6.1 丢弃高度相关的变量

[首先，如果两个变量高度相关，我们会删除其中一个变量。为了找到这些相关的变量对，我再次使用了相关矩阵（见第6.1节）。例如：GarageCars和GarageArea的相关系数为0.89。在这两个变量中，我会删除与SalePrice相关性最低的变量（GarageArea，与SalePrice的相关系数为0.62）。GarageCars与SalePrice的相关系数为0.64](https://zhuanlan.zhihu.com/p/136771737" \t "https://www.bing.com/_blank)

1. dropVars <- c('YearRemodAdd', 'GarageYrBlt', 'GarageArea', 'GarageCond', 'TotalBsmtSF', 'TotalRmsAbvGrd', 'BsmtFinSF1')
2. all <- all[,!(names(all) %in% dropVars)]

6.2 去除离群值

1. all <- all[-c(524, 1299),]#记住1299，这对之后修改Test\_Label非常重要

6.3 对变量进行预处理

6.3.1 数字型变量的中心化和缩放

将其数字型变量进行中心化，并为分类预测变量创建虚拟变量。将数据框拆分为一个包含所有数字变量的Dataframe和一个包含序数因子的Dataframe。

1. *#拆分*
2. numericVarNames <- numericVarNames[!(numericVarNames %in% c('MSSubClass', 'MoSold', 'YrSold', 'SalePrice', 'OverallQual', 'OverallCond'))] *#numericVarNames was created before having done anything*
3. numericVarNames <- append(numericVarNames, c('Age', 'TotalPorchSF', 'TotBathrooms', 'TotalSqFeet'))
4. DFnumeric <- all[, names(all) %in% numericVarNames]
5. DFfactors <- all[, !(names(all) %in% numericVarNames)]
6. DFfactors <- DFfactors[, names(DFfactors) != 'SalePrice']
7. cat('There are', length(DFnumeric), 'numeric variables, and', length(DFfactors), 'factor variables')
8. *#中心化*
9. for(i in 1:ncol(DFnumeric)){
10. if (abs(skew(DFnumeric[,i]))>0.8){
11. DFnumeric[,i] <- log(DFnumeric[,i] +1)
12. }
13. }
14. *#标准化*
15. PreNum <- preProcess(DFnumeric, method=c("center", "scale"))
16. print(PreNum)
17. *#拼接*
18. DFnorm <- predict(PreNum, DFnumeric)
19. dim(DFnorm)

6.3.2 类别变量的独热编码

1. DFdummies <- as.data.frame(model.matrix(~.-1, DFfactors))
2. dim(DFdummies)

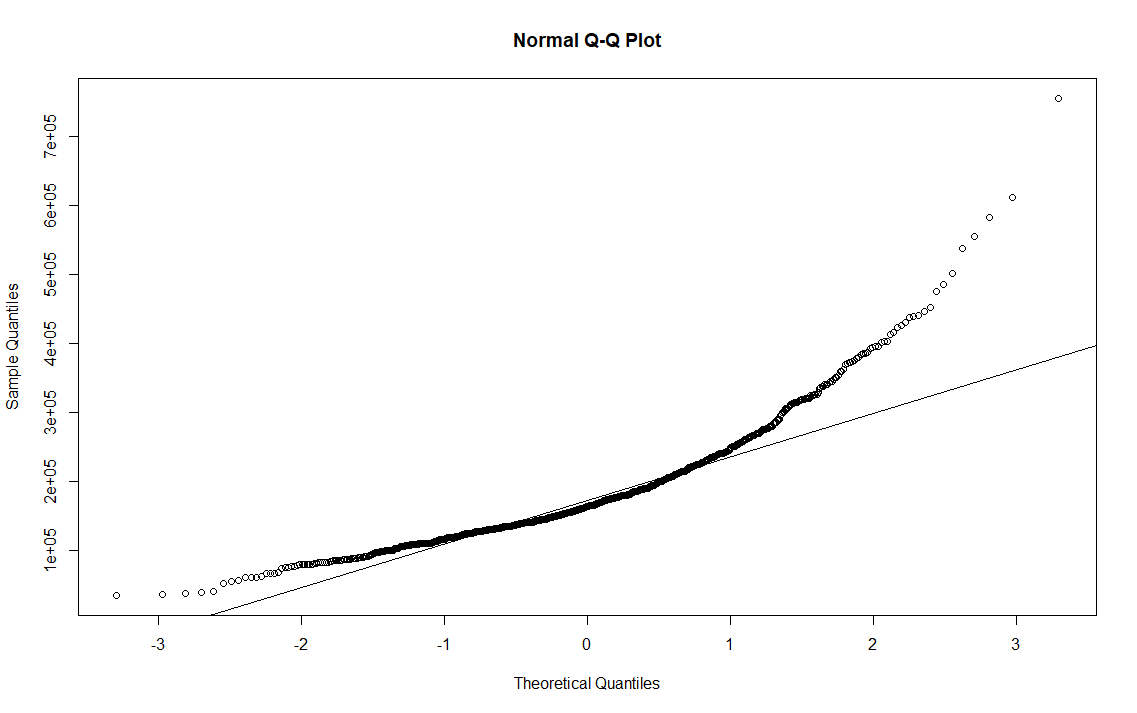
6.3.3去除一些个案数比较少的类别

完成独热编码后，我们需要去除一些个案数为0或者比较少(小于10)的属性

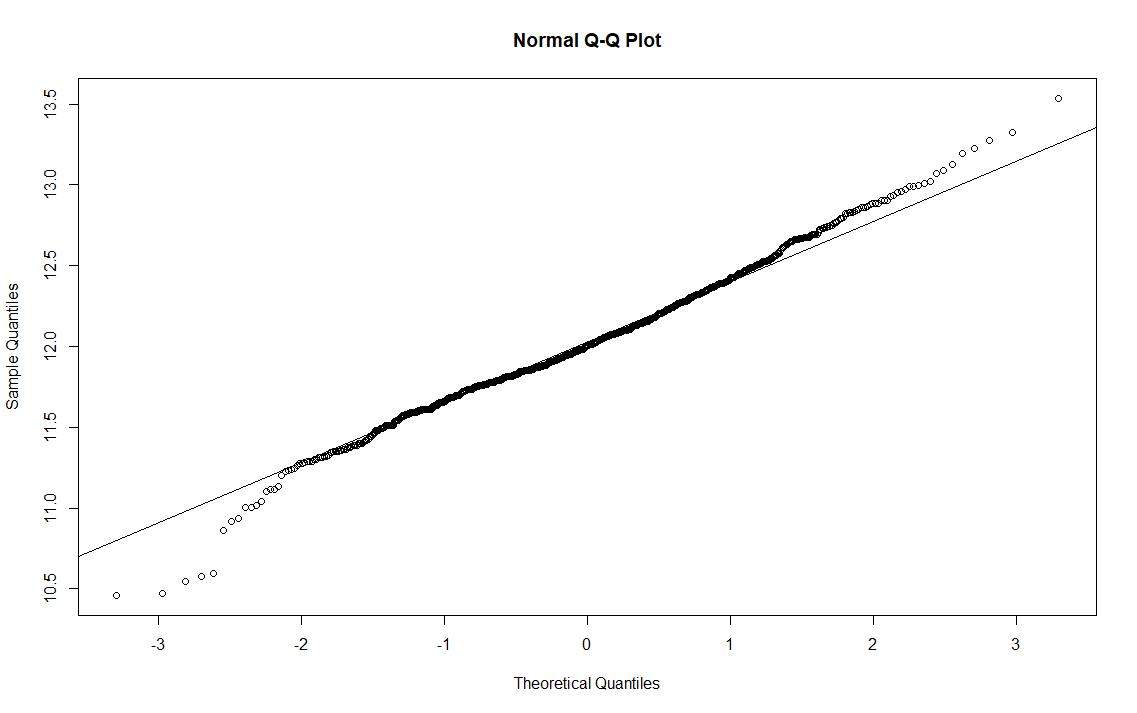
1. ZerocolTest <- which(colSums(DFdummies[(nrow(all[!is.na(all$SalePrice),])+1):nrow(all),])==0)
2. colnames(DFdummies[ZerocolTest])
3. DFdummies <- DFdummies[,-ZerocolTest] *#removing predictors*
4. ZerocolTrain <- which(colSums(DFdummies[1:nrow(all[!is.na(all$SalePrice),]),])==0)
5. colnames(DFdummies[ZerocolTrain])
6. DFdummies <- DFdummies[,-ZerocolTrain] *#removing predictor*
7. fewOnes <- which(colSums(DFdummies[1:nrow(all[!is.na(all$SalePrice),]),])<10)
8. colnames(DFdummies[fewOnes])
9. DFdummies <- DFdummies[,-fewOnes] *#removing predictors*
10. dim(DFdummies)
11. combined <- cbind(DFnorm, DFdummies) *#combining all (now numeric) predictors into one dataframe*

6.4 处理目标变量的偏度

1. skew(all$SalePrice)
2. qqnorm(all$SalePrice)
3. qqline(all$SalePrice)
4. all$SalePrice <- log(all$SalePrice) *#default is the natural logarithm, "+1" is not necessary as there are no 0's*
5. skew(all$SalePrice)
6. qqnorm(all$SalePrice)
7. qqline(all$SalePrice)



原始目标变量偏度较大。



对原始目标变量取对数，然后绘制QQ图，偏度被抑制。

6.5 得到最终的目标属性

1. train1 <- combined[!is.na(all$SalePrice),]
2. test1 <- combined[is.na(all$SalePrice),]

7 建模

7.1 XGBoost

7.1.1 定义参数网格

我们需要确定最优的训练参数，包括nrounds（迭代次数），eta（学习率），max\_depth（树的最大深度），gamma（节点分裂所需的最小损失减少），colsample\_bytree（每棵树使用的特征比例），min\_child\_weight（子节点的最小权重和），subsample（每棵树使用的样本比例）。这些参数会影响模型的复杂度和泛化能力，需要通过交叉验证来选择最优的组合。(运行时间非常长)

1. #xgb\_caret <- train(x=train1, y=all$SalePrice[!is.na(all$SalePrice)], method='xgbTree', trControl= my\_control, tuneGrid=xgb\_grid)
2. #xgb\_caret$bestTune

7.1.2 最优参数

我们确定的最优参数组合为

|  |  |
| --- | --- |
| max\_depth | 3 |
| eta | 0.05 |
| min\_child\_weight | 4 |
| nrounds | 341 |

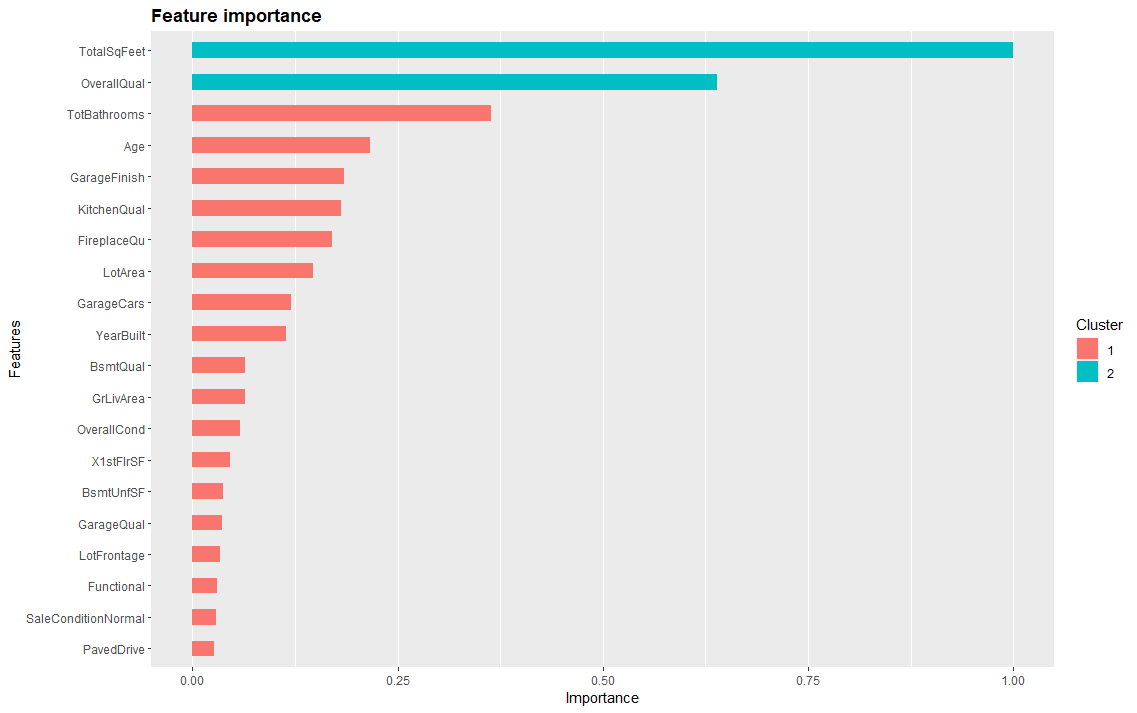
7.1.3 训练与评估

特别强调的是我们在数据处理时将第1299条去除了，为此我们需要将测试集中的第277条(1299-1022-277)去除，然后进行模型的评估。

1. xgb\_mod <- xgb.train(data = dtrain, params=default\_param, nrounds = 341)
2. XGBpred <- predict(xgb\_mod, dtest)
3. predictions\_XGB <- exp(XGBpred) *#need to reverse the log to the real values*
4. test\_label <- test\_label[-277] *#测试集没有，在标签中也去除这一个元素*
5. sqrt(mean((predictions\_XGB - test\_label)^2))
6. head(predictions\_XGB)
7. library(Ckmeans.1d.dp) *#required for ggplot clustering*
8. mat <- xgb.importance (feature\_names = colnames(train1),model = xgb\_mod)
9. xgb.ggplot.importance(importance\_matrix = mat[1:20], rel\_to\_first = TRUE)

|  |  |
| --- | --- |
| 训练参数 | 值 |
| objective | reg:linear |
| booster | gbtree |
| eta | 0.05 |
| gamma | 0 |
| max\_depth | 3 |
| min\_child\_weight | 4 |
| subsample | 1 |
| colsample\_bytree | 1 |

最终得到RMSE=24663.35，并绘制得到如下的变量重要性排名。



7.2 Lasso Regression

7.2.1定义参数网格

我们同样用5折交叉验证作为训练控制，参数网格包括alpha值和lambda

1. set.seed(27042018)
2. my\_control <-trainControl(method="cv", number=5)
3. lassoGrid <- expand.grid(alpha = 1, lambda = seq(0.001,0.1,by = 0.0005))
4. lasso\_mod <- train(x=train1, y=all$SalePrice[!is.na(all$SalePrice)], method='glmnet', trControl= my\_control, tuneGrid=lassoGrid)
5. lasso\_mod$bestTune

7.2.2 最优参数

Alpha=1，lambda=0003

7.2.3 训练参数与评估

1. lassoVarImp <- varImp(lasso\_mod,scale=F)
2. lassoImportance <- lassoVarImp$importance
3. varsSelected <- length(which(lassoImportance$Overall!=0))
4. varsNotSelected <- length(which(lassoImportance$Overall==0))
5. cat('Lasso uses', varsSelected, 'variables in its model, and did not select', varsNotSelected, 'variables.')
6. LassoPred <- predict(lasso\_mod, test1)
7. predictions\_lasso <- exp(LassoPred) *#need to reverse the log to the real values*
8. sqrt(mean((predictions\_lasso - test\_label)^2))

|  |  |
| --- | --- |
| 训练参数 | 值 |
| method | glmnet |

最终训练得到RMSE=23828.68，使用了46%(80/173)的变量进行回归。

7.3 随机森林

7.3.1 模型参数

1. set.seed(2018)
2. quick\_RF <- randomForest(x=train1, y=all$SalePrice[!is.na(all$SalePrice)], ntree=100,importance=TRUE)
3. RFPred <- predict(quick\_RF, test1)
4. predictions\_RF <- exp(RFPred)
5. sqrt(mean((predictions\_RF - test\_label)^2))

直接使用默认参数ntree=100

7.3.2 训练与评估

模型得到的RMSE=27463.56，并绘制得到重要性排名。

