House Prices: Advanced Regression Techniques

Kaggle Compeition 1 for DA

Sean Duan

10/26/2020

Introduction

This problem is a relatively simple one, determining which combined aspects of a property determine it's value on the open market. Our dataset is composed of residential home information from Ames, Iowa.

Preliminary Data Analysis

The first thing I did was decide what elements were important in order to have 'clean' data to work from. I made the decision to drop columns with NA values from the dataset, as well as removing any columns with perfect collinearity with other predictors in the dataset. Additionally, many of our factorial predictors had factors with extremely small numbers of entries, sometimes as small as a single entry. I found these entries by running a For loop that looked for numbers of entry in each level of a factor for all the factorial predictors. To solve this issue, I engaged in some feature engineering.

Primarily, I chose to subset smaller categories into larger ones. While details on exactly which selections I made can be found below in my code annotations, I briefly explain a few select issues ahead. For things that had a clear ordinal organization, I collapsed a given entry into it's next nearest category. For example, collapsing our category of poor heating quality, into it's next nearest, 'fair' heating quality. For some of our factors, no ordinal grouping existed. For the groups that simply had no similarities, for example, our 'foundation' factor, since wood and stone had very few entries, I collapsed them into a 'other' category. For categories that simply were subsets of larger potential categories, such as the 'saletype' predictor, I chose to collapse all the various warrantee deeds into one selection, and all of the various contract types into one selection.

Lastly, to prepare the test data, I ensured that all elements were the same as in the training data, and imputed missing values. The missing value was either 0 in a numerical category, or was the most common category in a factorial category.

Plots

To help guide my intution with regards as to which elements were related to house pricing, I did a simple plot that compared house prices to our various predictors. The variables that stood out to me as being highly correlated, and worth paying attention to down the road were sale type, sale condition, gr liv area, foundation, exterior 1st, exter qual, bldgtype, housestyle, neighborhood, mssubclass, and mszoning.

Model Type Selection

I used 2-fold Cross Validation, as there were still enough rare categories that I was not able to get satisfactory stratified sampling throughout more than 2 folds on the training data. I chose the models that had the best performance on two fold cross validation through the average mean square error for all the folds combined.

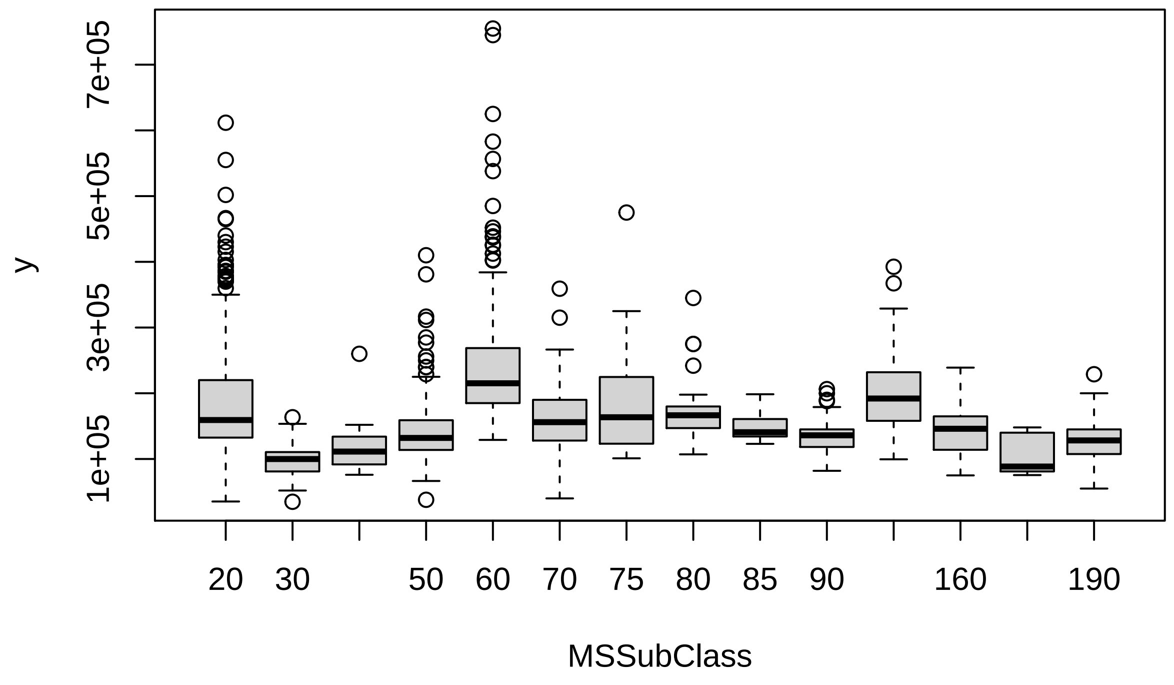
However, I was unable to get the test data to recognize that it had the exact same factor names as the training data, and thus the random forest model (which had the lowest training mse in our 2 fold CV) and bagging were unable to be run on our test data.

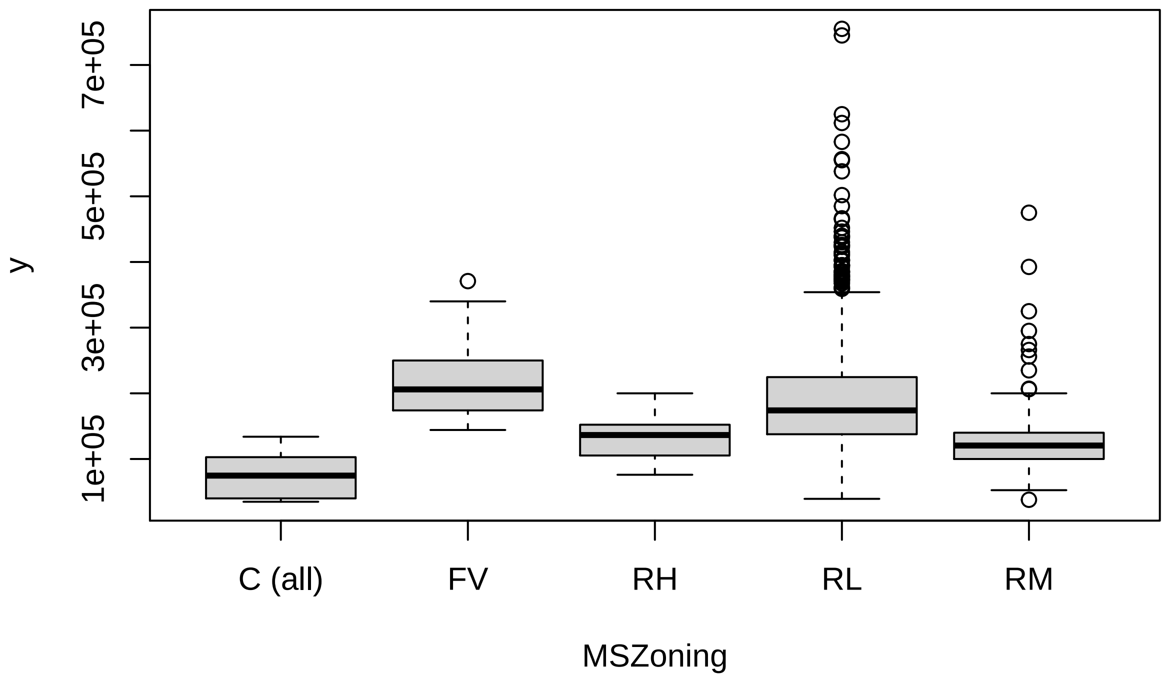
Data Cleaning Code

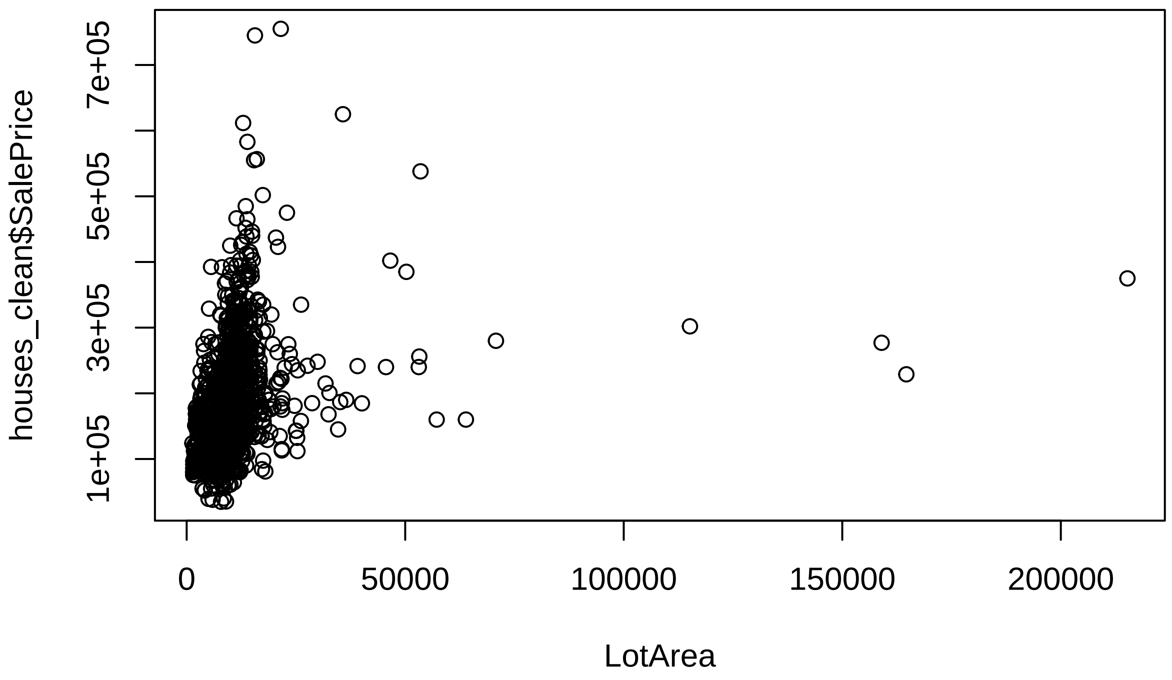
```r
#data loading, prep, and cleaning step.
houses<-read.csv("train.csv")
houses_test<-read.csv("test.csv")
houses$MSSubClass<-as.factor(houses$MSSubClass)
houses_nna<-houses %>%select_if(~ !any(is.na(.)))
houses_clean<-subset(houses_nna, select=-c(X1stFlrSF, X2ndFlrSF, LowQualFinSF, BsmtFinSF1, BsmtFinSF2, 


#ok, can't do k-fold b/c we have lack of uniqueness in railroad
#time for feature engineering - collapse the railroad categories
#and positive benefit categories
houses_clean$Condition2<-fct_recode(houses_clean$Condition2, AF = "Artery", AF = "Feedr", RR = "RRAe", 
houses_clean$Condition1<-fct_recode(houses_clean$Condition1, AF = "Artery", AF = "Feedr", RR = "RRAe", 
#mesh roof type to composite/ not composite
houses_clean$RoofMatl<-fct_recode(houses_clean$RoofMatl, Oth = "ClyTile", Oth = "Membran", Oth = "Metal
#collapse shingles, bricks/block/sstone, imstucco w/ stucco,
houses_clean$Exterior1st<-fct_recode(houses_clean$Exterior1st, Shng = "AbShng",Shng = "AsphShn",Shng = 
                                     Brx = "BrkFace", Brx = "CBlock",Brx = "Stone",Stuc = "Stucco",Stuc
#collapse poor and fair
houses_clean$ExterCond<-fct_recode(houses_clean$ExterCond , PF = "Po", PF = "Fa", NULL = "H")
#collapse heating, not gas
houses_clean$Heating<-fct_recode(houses_clean$Heating , Oth = "Floor", Oth = "Grav", Oth = "OthW", Oth 
#collapse poor and fair
houses_clean$HeatingQC<-fct_recode(houses_clean$HeatingQC , PF = "Po", PF = "Fa", NULL = "H")

#collapse contract terms, warranties deeds
houses_clean$SaleType<-fct_recode(houses_clean$SaleType ,Cont = "Oth", Cont = "Con", Cont = "ConLD", Co

#collapse bluestem to NAmes(most similar)
houses_clean$Neighborhood<-fct_recode(houses_clean$Neighborhood , NAmesP = "Blueste", NAmesP = "NAmes", 

#collapse roof type shd w/ flat
houses_clean$RoofStyle<-fct_recode(houses_clean$RoofStyle , FS = "Flat", FS = "Shed", NULL = "H")

#collapse stone w/ wood for 'othr' foundatain
houses_clean$Foundation<-fct_recode(houses_clean$Foundation , Oth = "Stone", Oth = "Wood", NULL = "H")

#collapse major 1-2 and severe
houses_clean$Functional<-fct_recode(houses_clean$Functional , MSev = "Maj1", MSev = "Maj2", MSev = "Sev

#collapse frontage
houses_clean$LotConfig<-fct_recode(houses_clean$LotConfig , FR = "FR2", FR = "FR3", NULL = "H")

#collapse mssubclass 40 and 45
houses_clean$MSSubClass<-fct_recode(houses_clean$MSSubClass , "42.5" = "40", "42.5" = "45", NULL = "H")

#make sure all our characters are factors
```

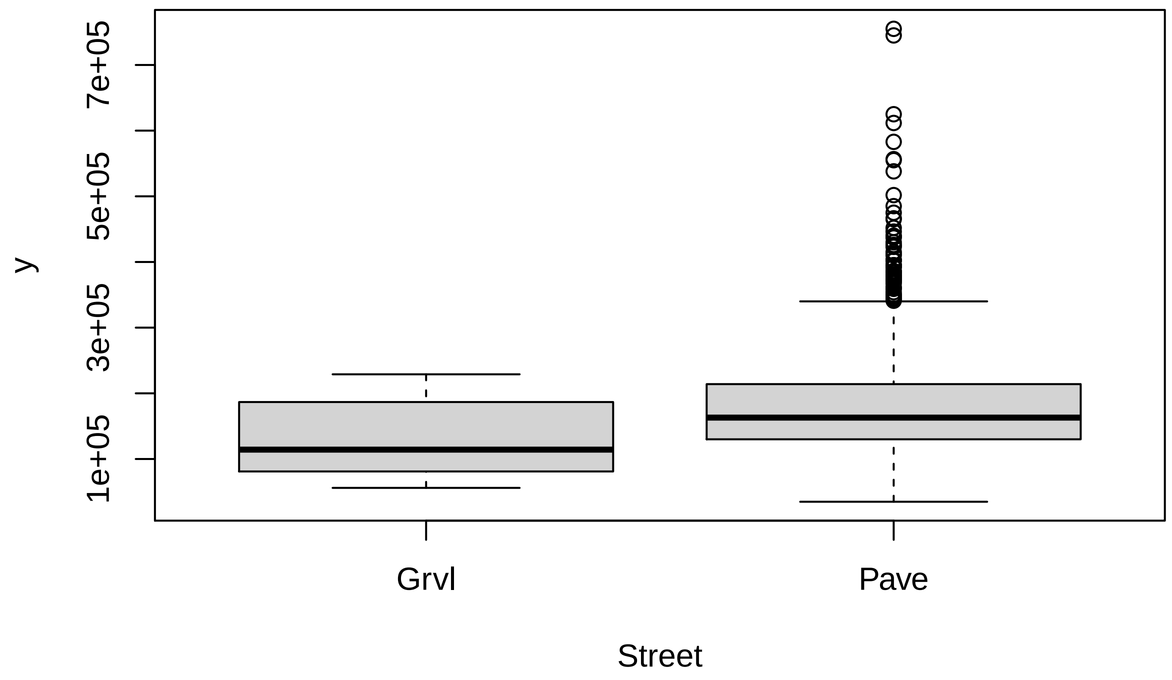
```
houses_clean <- houses_clean %>% mutate_if(is.character,as.factor)




#code to prep houses-test
varnames<-names(houses_clean)
houses_clean_tst<-houses_test
houses_clean_tst<-houses_clean_tst[ names(houses_clean_tst)[names(houses_clean_tst) %in% varnames] ]

#time for feature engineering - collapse the railroad categories
#and positive benefit categories
houses_clean_tst$Condition2<-fct_recode(houses_clean_tst$Condition2, AF = "Artery", AF = "Feedr", RR =
houses_clean_tst$Condition1<-fct_recode(houses_clean_tst$Condition1, AF = "Artery", AF = "Feedr", RR =
#mesh roof type to composite/ not composite
houses_clean_tst$RoofMatl<-fct_recode(houses_clean_tst$RoofMatl, Oth = "ClyTile", Oth = "Membran", Oth =
#collapse shingles, bricks/block/sstone, imstucco w/ stucco,
houses_clean_tst$Exterior1st<-fct_recode(houses_clean_tst$Exterior1st, Shng = "AbShng",Shng = "AsphShn"
                                     Brx = "BrkFace", Brx = "CBlock",Brx = "Stone",Stuc = "Stucco",Stuc
#collapse poor and fair
houses_clean_tst$ExterCond<-fct_recode(houses_clean_tst$ExterCond , PF = "Po", PF = "Fa", NULL = "H")
#collapse heating, not gas
houses_clean_tst$Heating<-fct_recode(houses_clean_tst$Heating , Oth = "Floor", Oth = "Grav", Oth = "OthW
#collapse poor and fair
houses_clean_tst$HeatingQC<-fct_recode(houses_clean_tst$HeatingQC , PF = "Po", PF = "Fa", NULL = "H")

#collapse contract terms, warranties deeds
houses_clean_tst$SaleType<-fct_recode(houses_clean_tst$SaleType ,Cont = "Oth", Cont = "Con", Cont = "Con

#collapse bluestem to NAmes(most similar)
houses_clean_tst$Neighborhood<-fct_recode(houses_clean_tst$Neighborhood , NAmesP = "Blueste", NAmesP =

#collapse roof type shd w/ flat
houses_clean_tst$RoofStyle<-fct_recode(houses_clean_tst$RoofStyle , FS = "Flat", FS = "Shed", NULL = "H

#collapse stone w/ wood for 'othr' foundatain
houses_clean_tst$Foundation<-fct_recode(houses_clean_tst$Foundation , Oth = "Stone", Oth = "Wood", NULL

#collapse major 1-2 and severe
houses_clean_tst$Functional<-fct_recode(houses_clean_tst$Functional , MSev = "Maj1", MSev = "Maj2", MSev

#collapse frontage
houses_clean_tst$LotConfig<-fct_recode(houses_clean_tst$LotConfig , FR = "FR2", FR = "FR3", NULL = "H")

#collapse mssubclass 40 and 45
houses_clean_tst$MSSubClass<-as.factor(houses_clean_tst$MSSubClass)
houses_clean_tst$MSSubClass<-fct_recode(houses_clean_tst$MSSubClass , "42.5" = "40", "42.5" = "45", "160


#imputing missing values
houses_clean_tst$MSZoning<-replace_na(houses_clean_tst$MSZoning, "RM")
houses_clean_tst$Exterior1st<-replace_na(houses_clean_tst$Exterior1st, "MetalSd")
houses_clean_tst$TotalBsmtSF<-replace_na(houses_clean_tst$TotalBsmtSF, 0)
```

```
houses_clean_tst$BsmtFullBath<-replace_na(houses_clean_tst$BsmtFullBath, 0)
houses_clean_tst$BsmtHalfBath<-replace_na(houses_clean_tst$BsmtHalfBath, 0)
houses_clean_tst$SaleType<-replace_na(houses_clean_tst$SaleType, "New")
houses_clean_tst$GarageCars<-replace_na(houses_clean_tst$GarageCars, 0)
houses_clean_tst$GarageArea<-replace_na(houses_clean_tst$GarageArea, 0)
houses_clean_tst$Functional<-replace_na(houses_clean_tst$Functional, "Typ")
houses_clean_tst$KitchenQual<-replace_na(houses_clean_tst$KitchenQual, "TA")


#make sure all our characters are factors
houses_clean_tst <- houses_clean_tst %>% mutate_if(is.character,as.factor)

for(i in 1:53){
  plot(y=houses_clean$SalePrice, x=houses_clean[,i], xlab=colnames(houses_clean)[i])
}
```

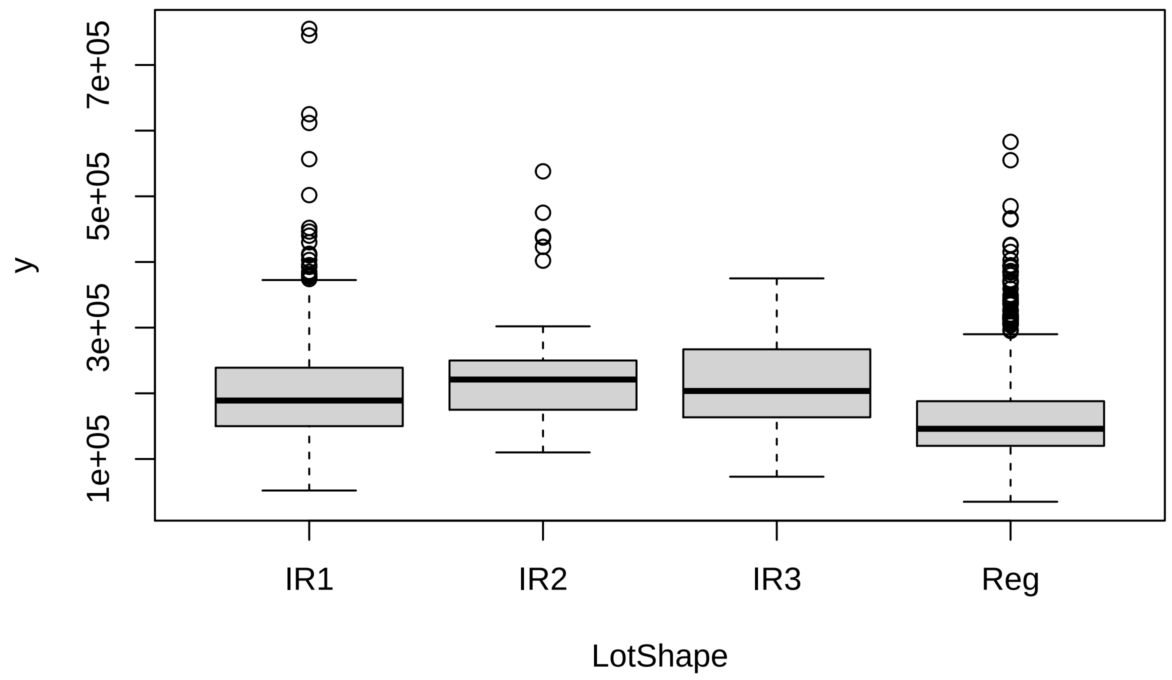


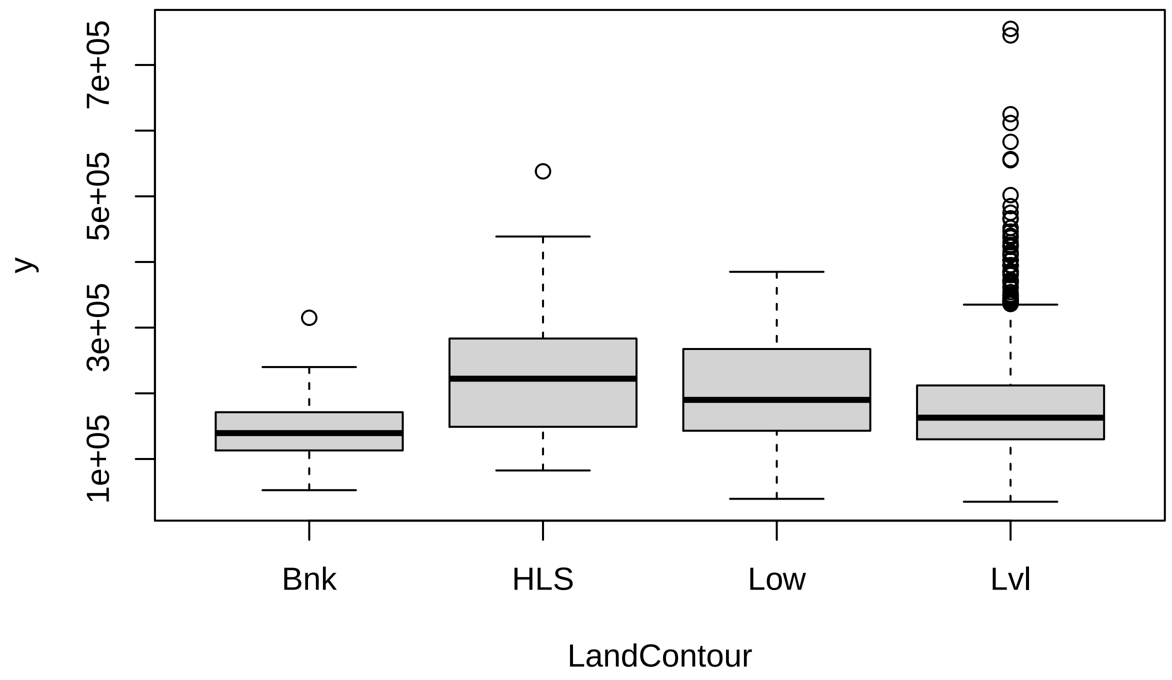

y
1e+05
3e+05
5e+05
7e+05
C (all)
FV
RH
RL
RM
MSZoning


houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
0
50000
100000
150000
200000
LotArea

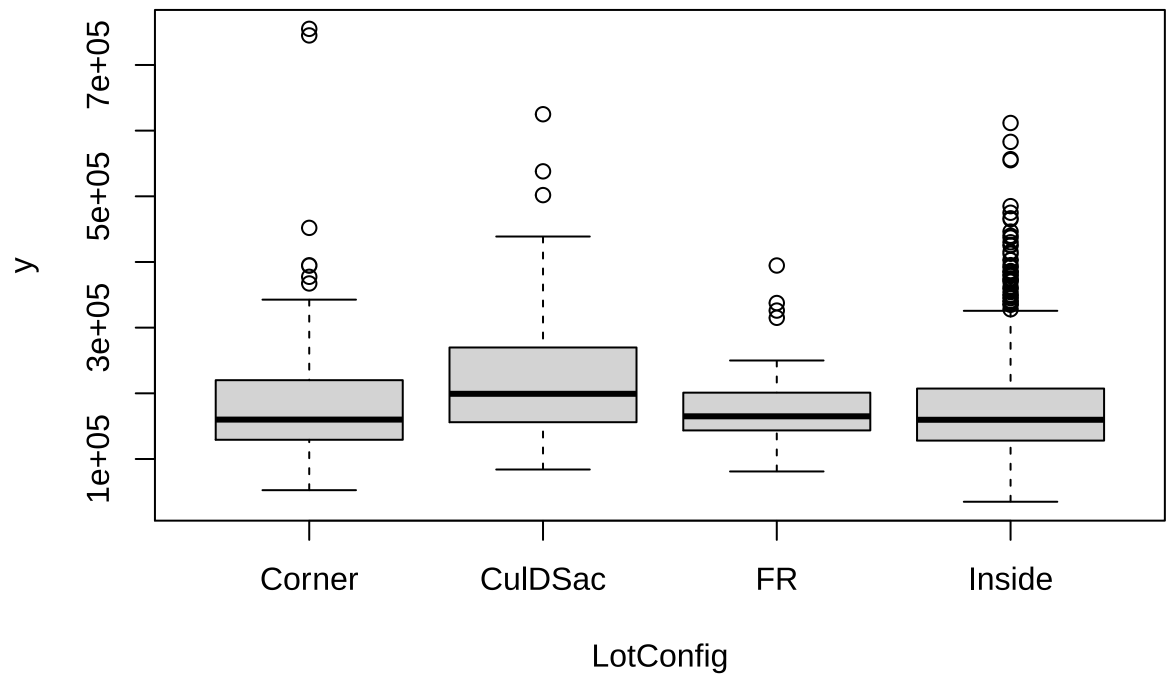


y
1e+05
3e+05
5e+05
7e+05
Grvl
Pave
Street

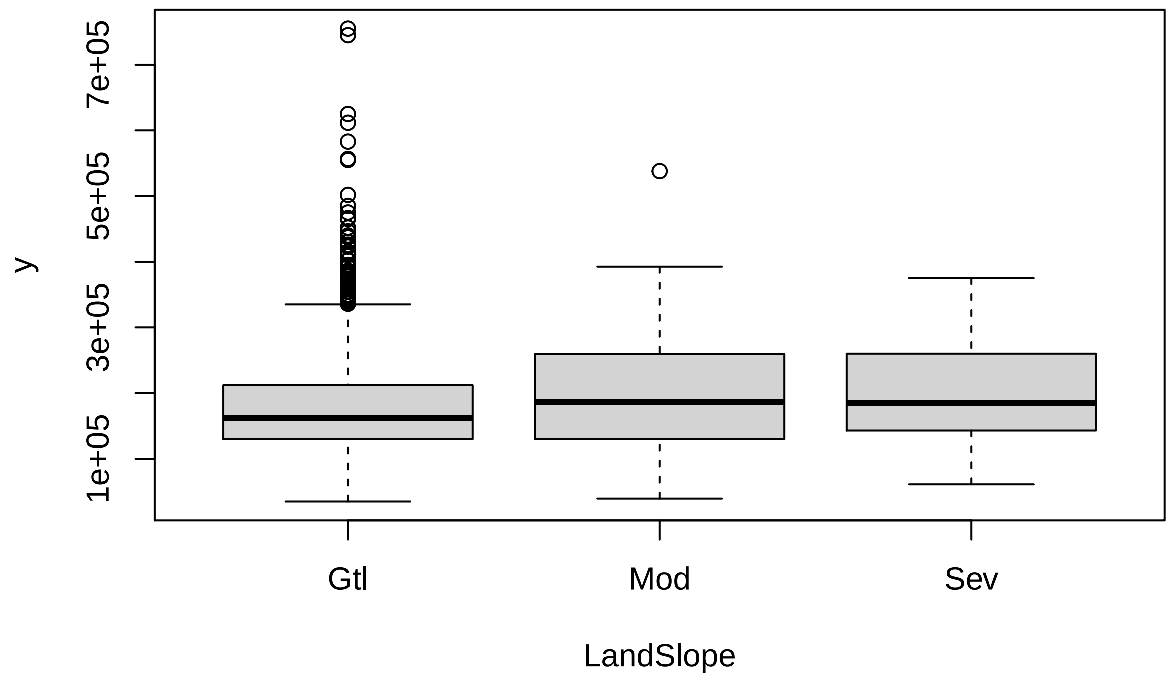

y
1e+05
3e+05
5e+05
7e+05
IR1
IR2
IR3
Reg
LotShape

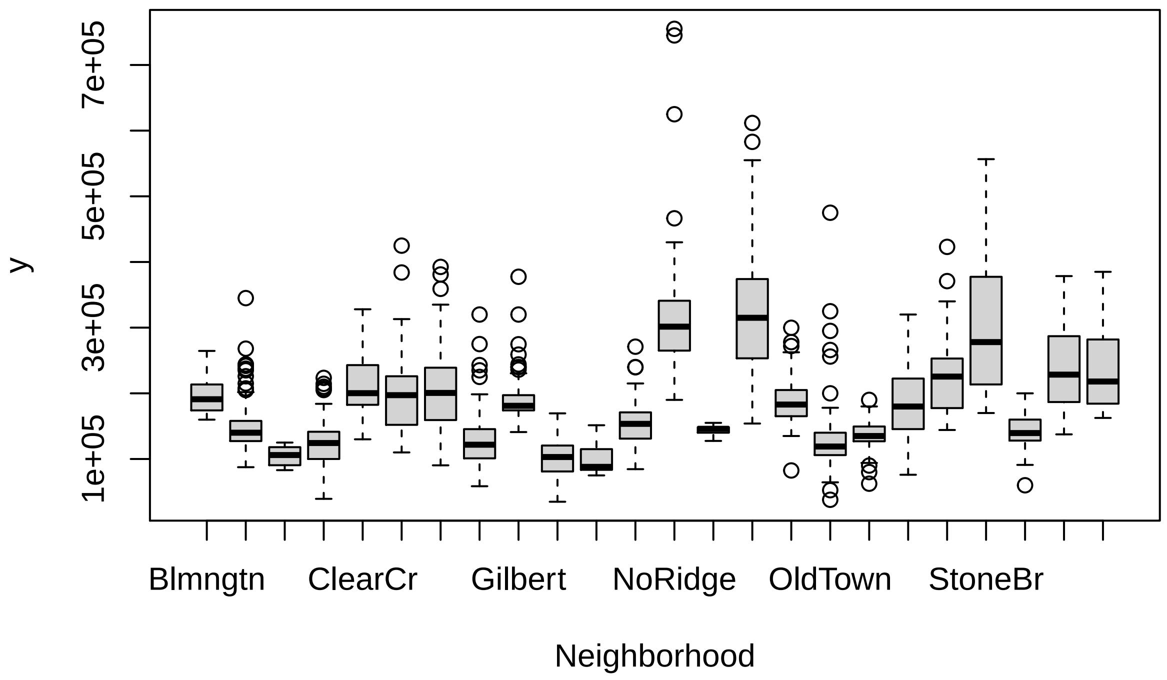


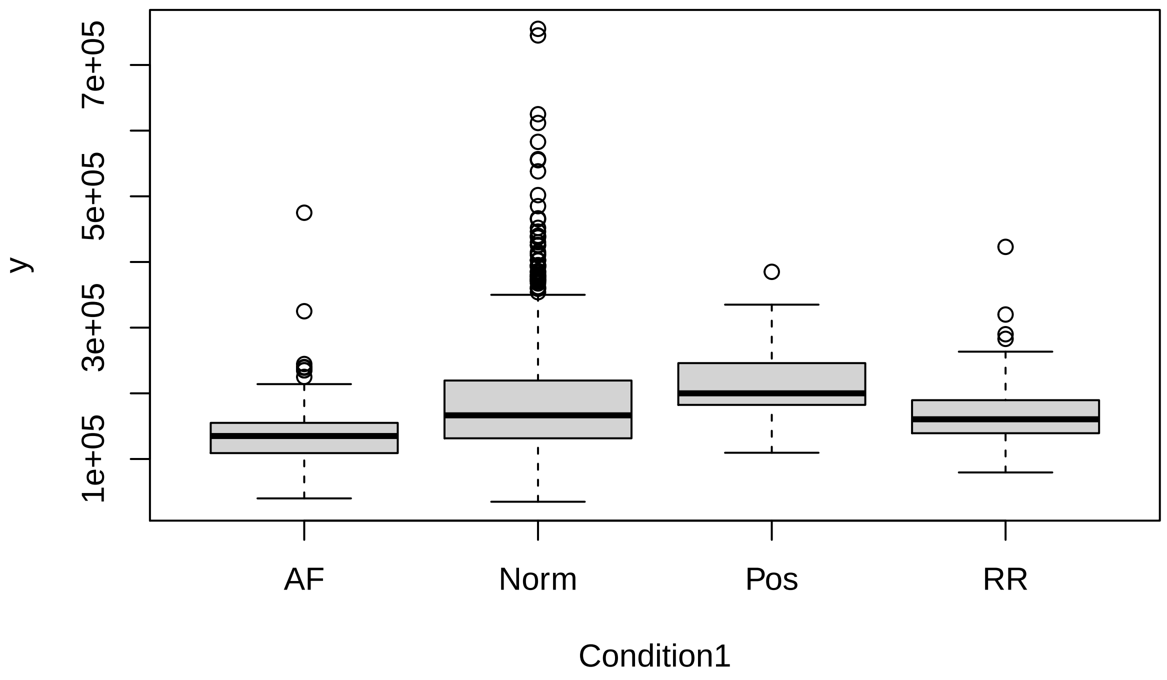
7e+05
5e+05
3e+05
1e+05
y
Bnk
HLS
Low
Lvl
LandContour



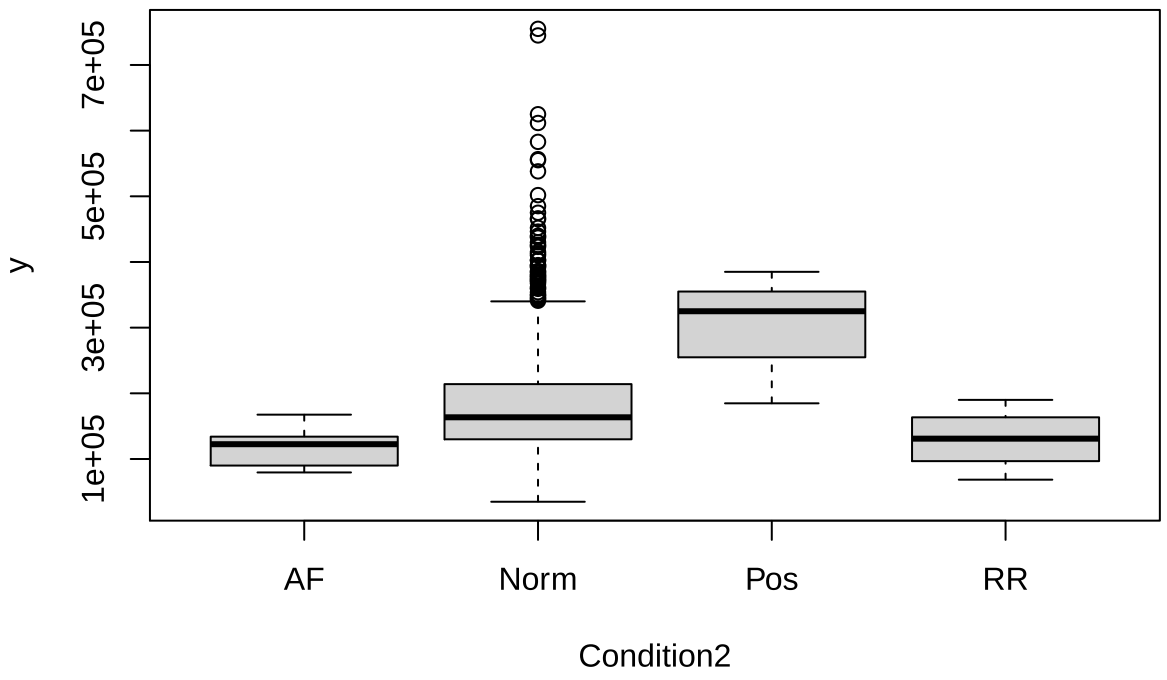
y
1e+05
3e+05
5e+05
7e+05
Corner
CulDSac
FR
Inside
LotConfig

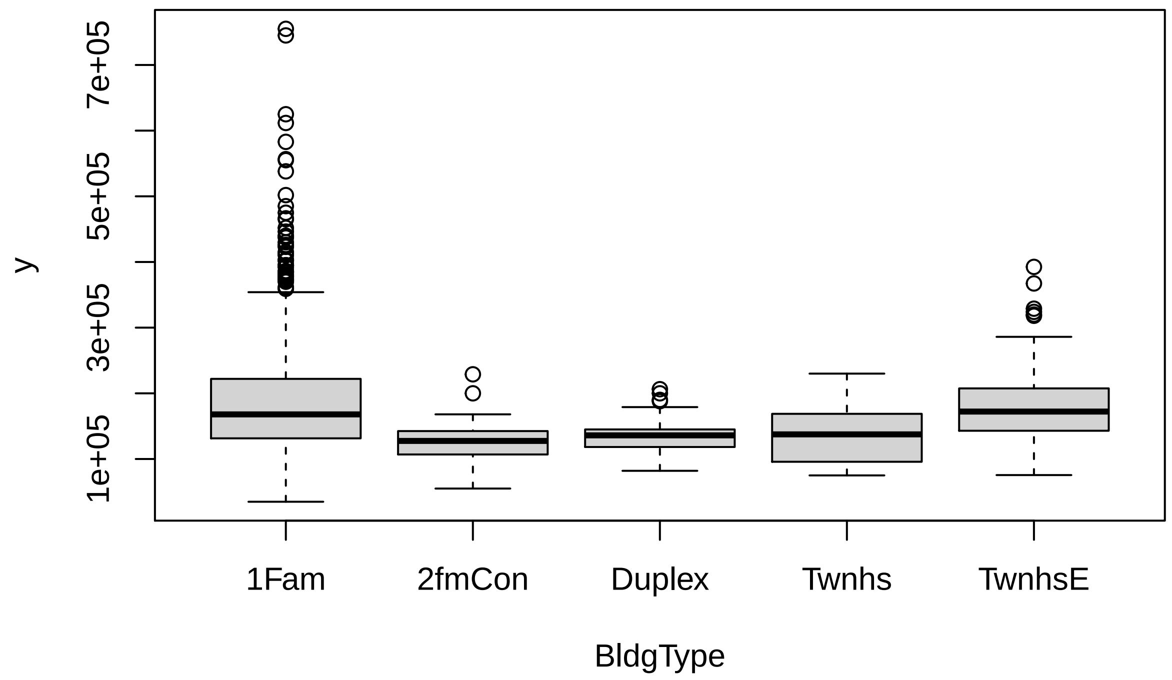

y
1e+05
3e+05
5e+05
7e+05
Gtl
Mod
Sev
LandSlope

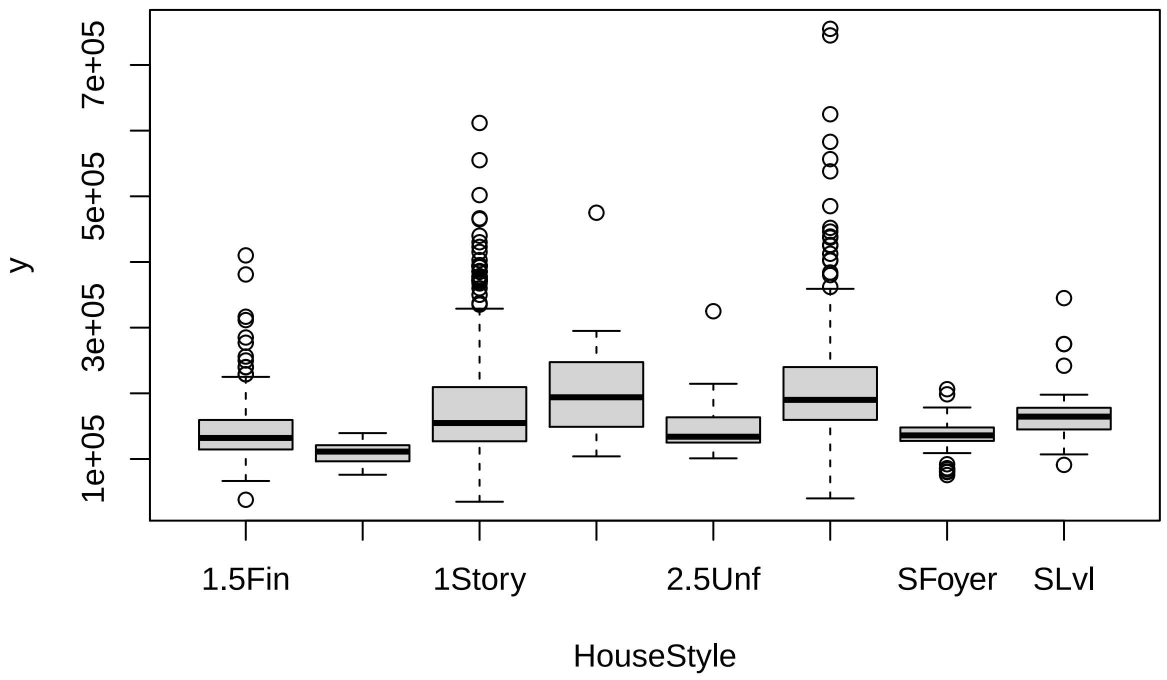

y
1e+05
3e+05
5e+05
7e+05
Blmngtn
ClearCr
Gilbert
NoRidge
OldTown
StoneBr
Neighborhood

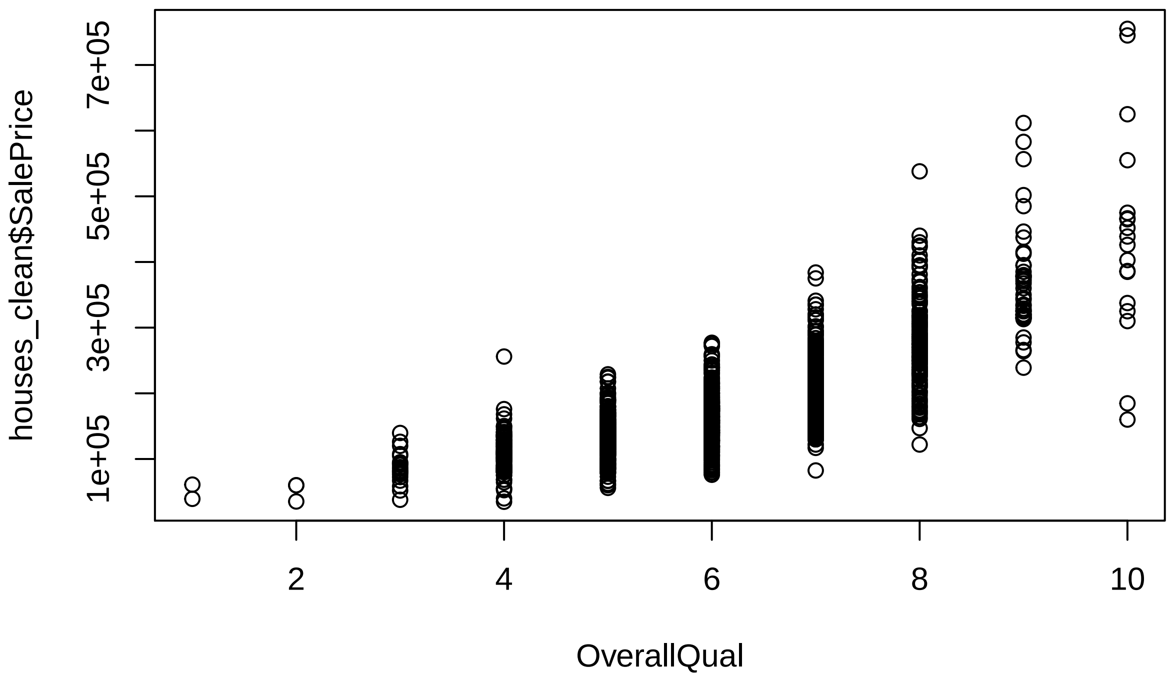


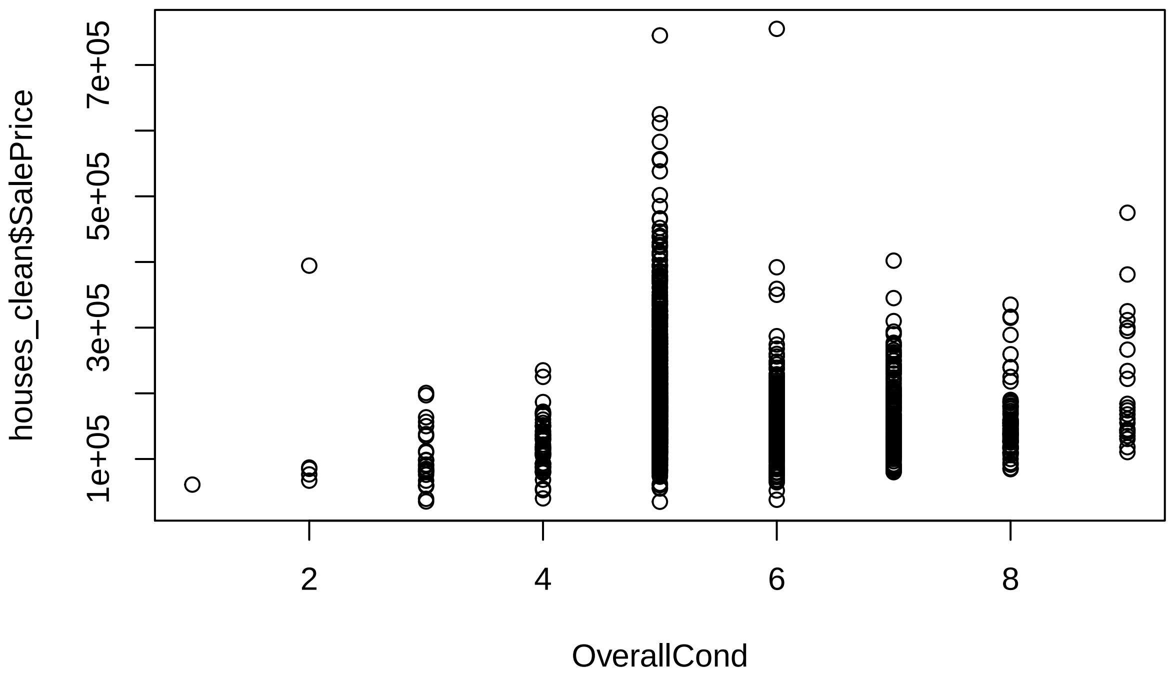
y
1e+05
3e+05
5e+05
7e+05
AF
Norm
Pos
RR
Condition1

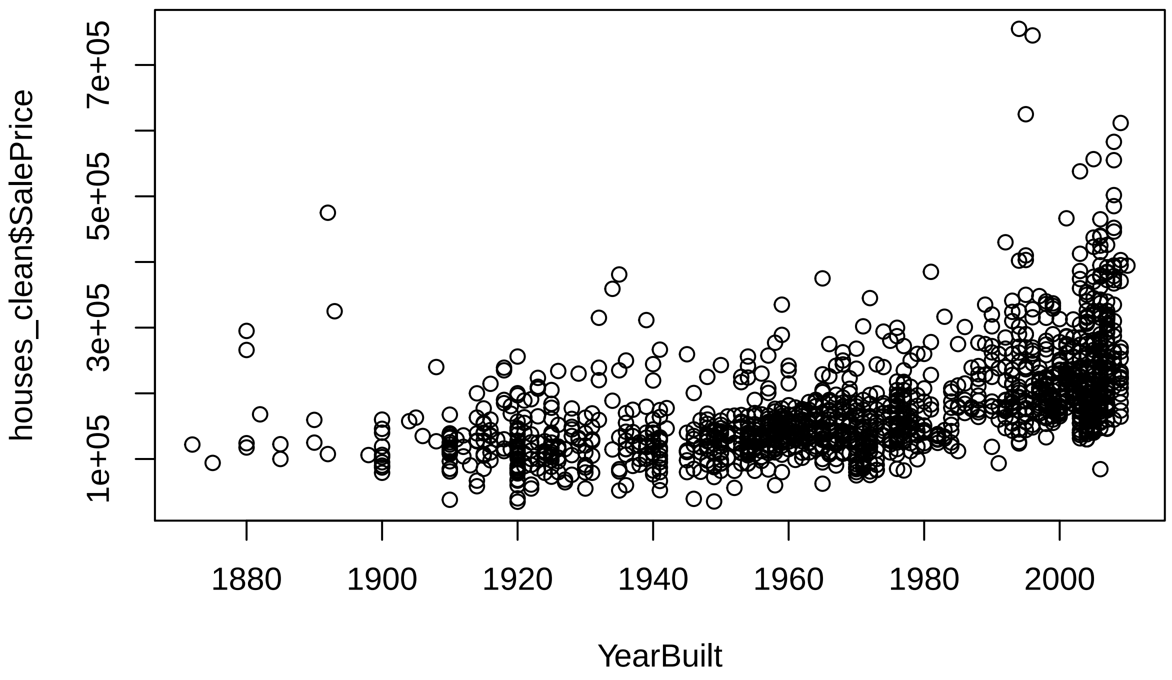

y
1e+05
3e+05
5e+05
7e+05
AF
Norm
Pos
RR
Condition2

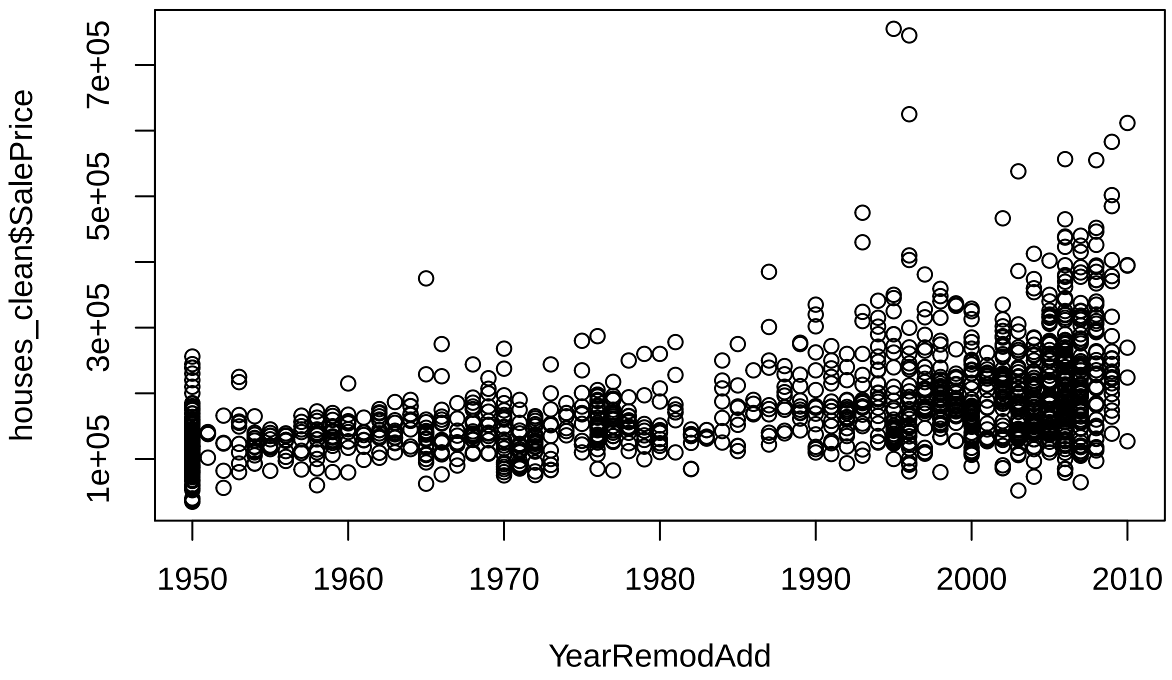

y
1e+05
3e+05
5e+05
7e+05
1Fam
2fmCon
Duplex
Twnhs
TwnhsE
BldgType

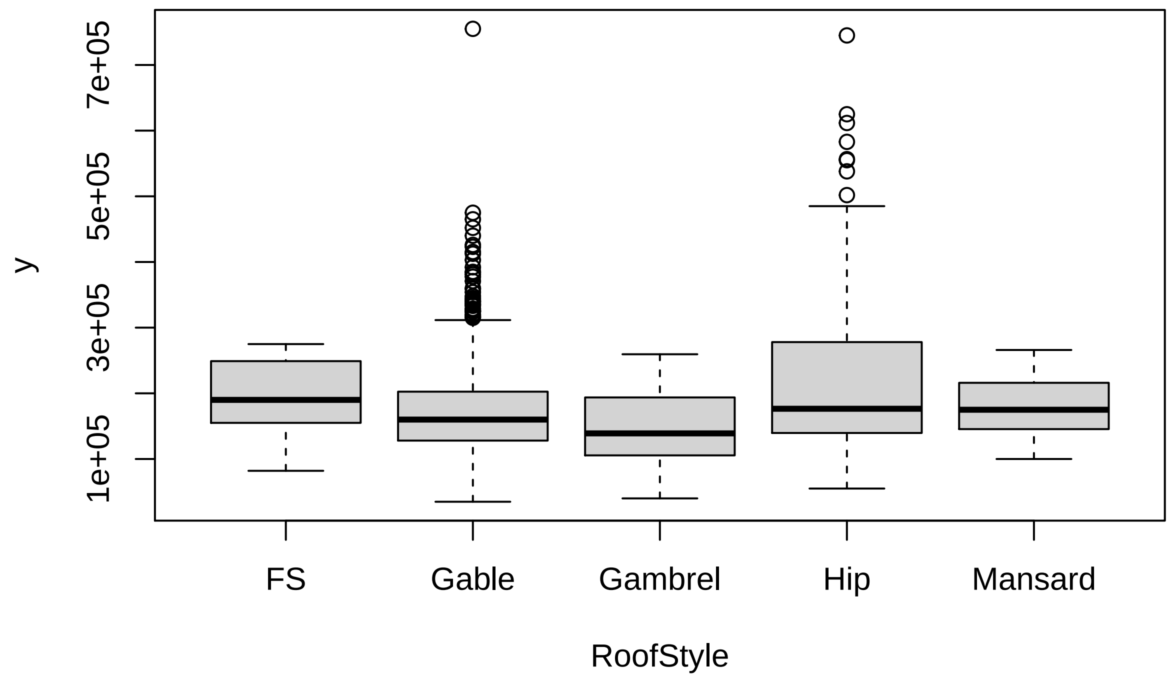

7e+05
5e+05
3e+05
1e+05
y
1.5Fin
1Story
2.5Unf
SFoyer
SLvl
HouseStyle

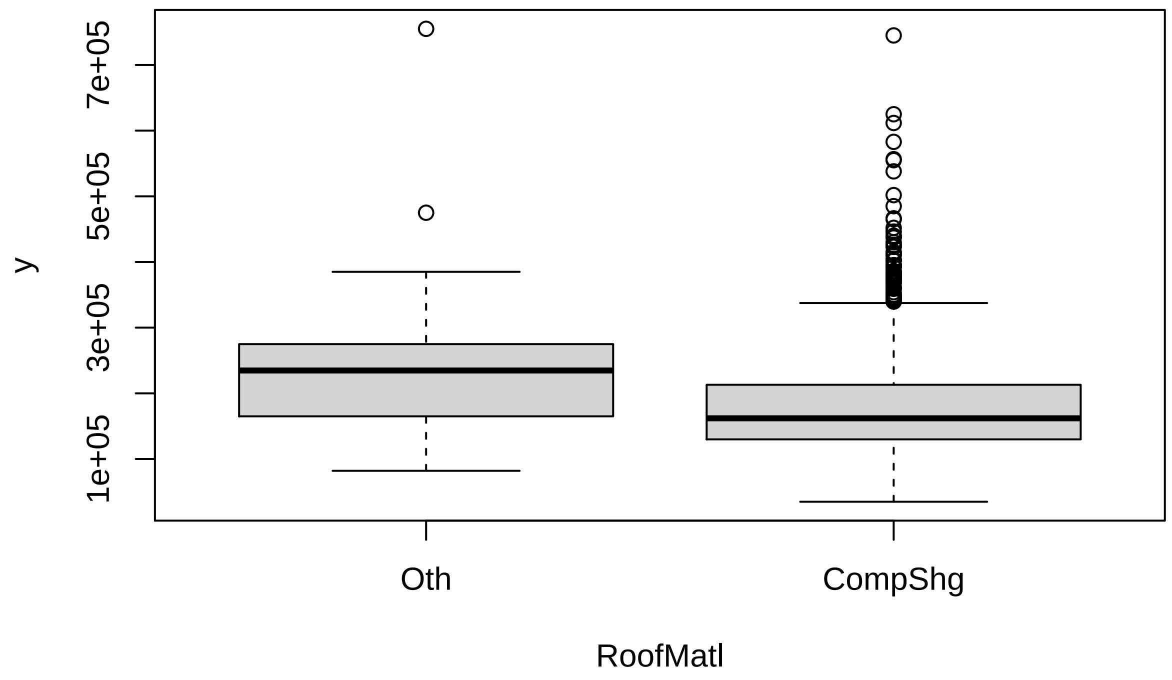

houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
2
4
6
8
10
OverallQual


houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
2
4
6
8
OverallCond

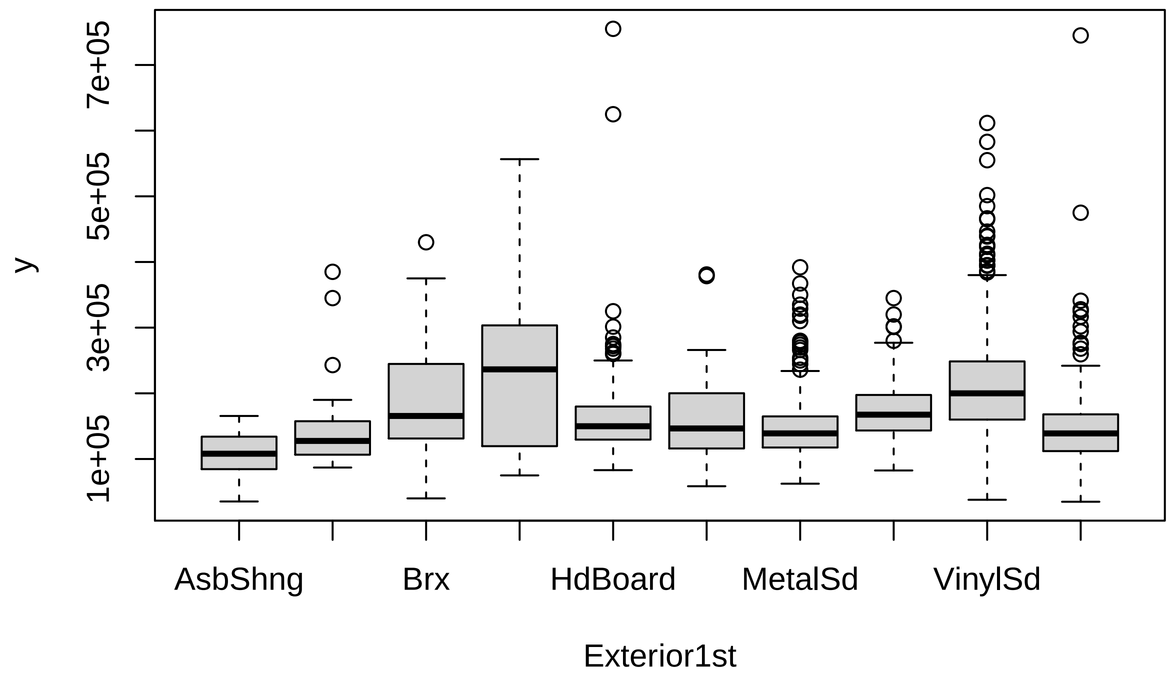

houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
1880
1900
1920
1940
1960
1980
2000
YearBuilt

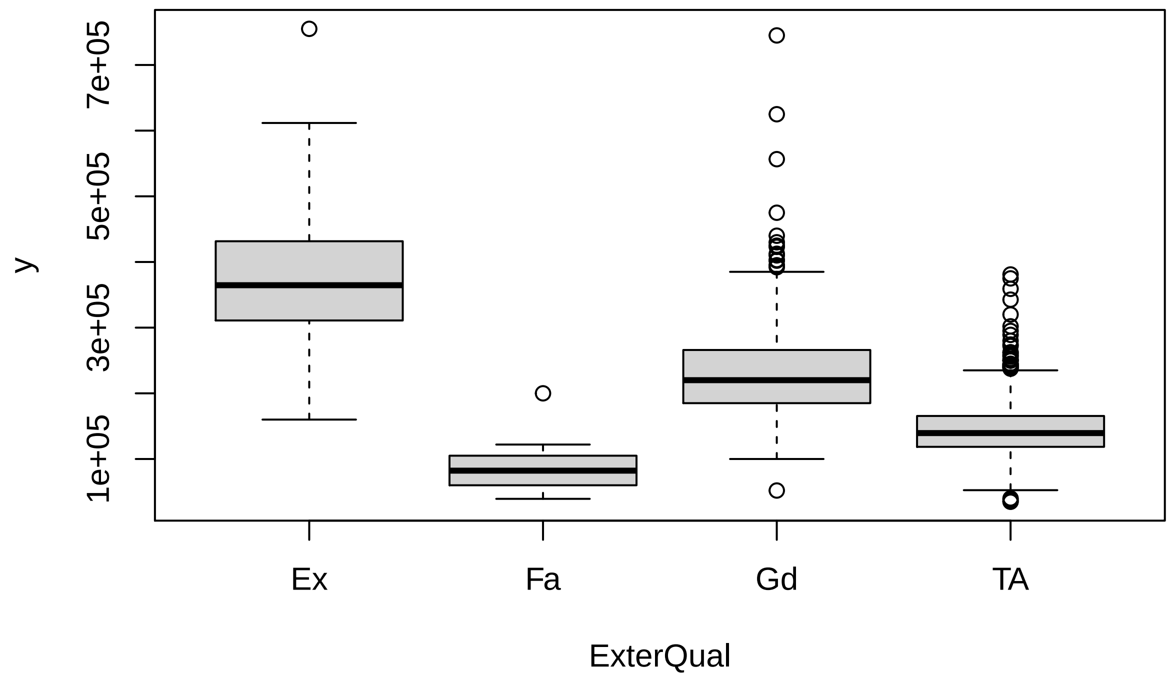

houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
1950
1960
1970
1980
1990
2000
2010
YearRemodAdd


y
1e+05
3e+05
5e+05
7e+05
FS
Gable
Gambrel
Hip
Mansard
RoofStyle

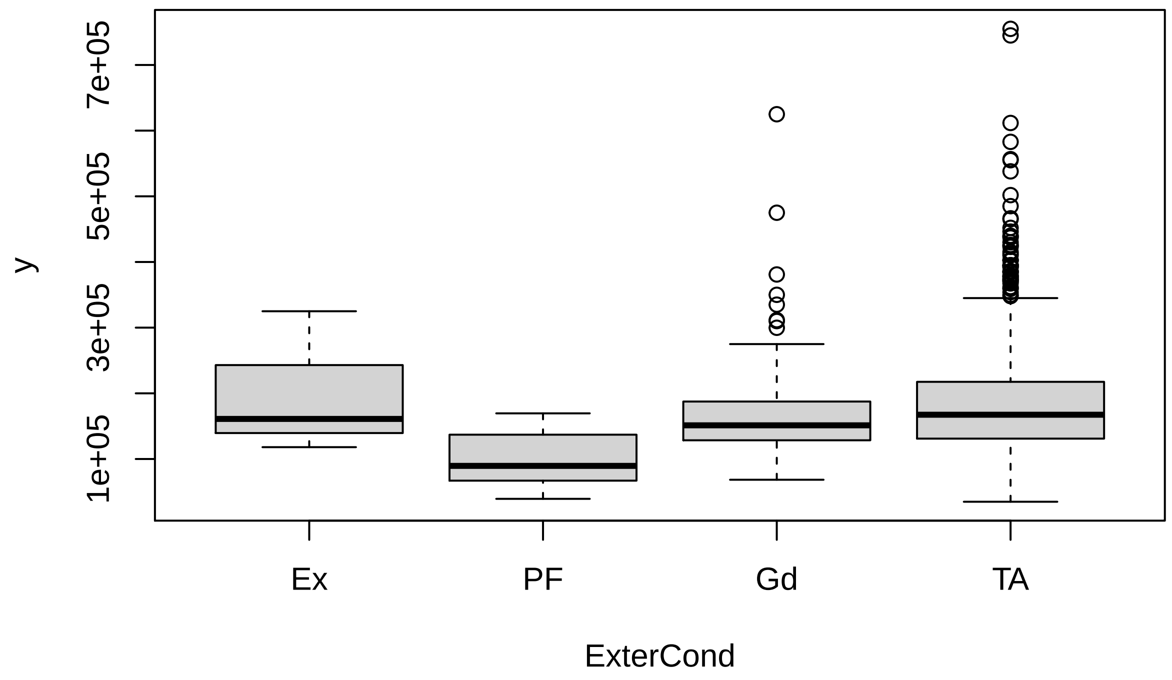


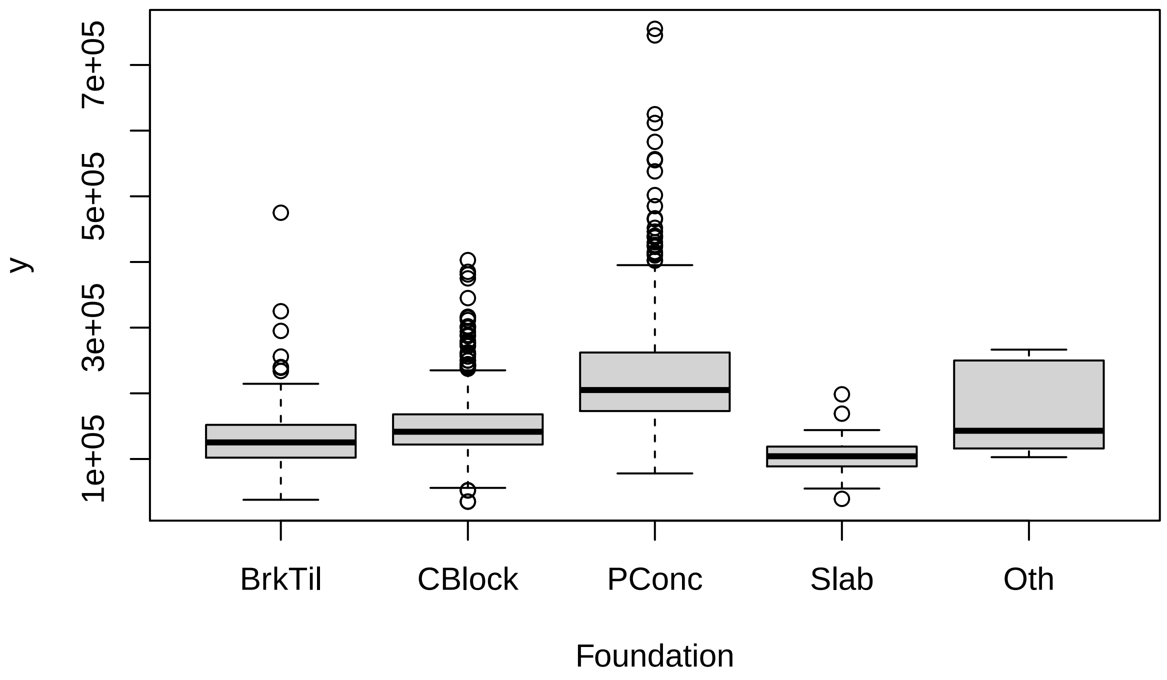
y
1e+05
3e+05
5e+05
7e+05
Oth
CompShg
RoofMatl

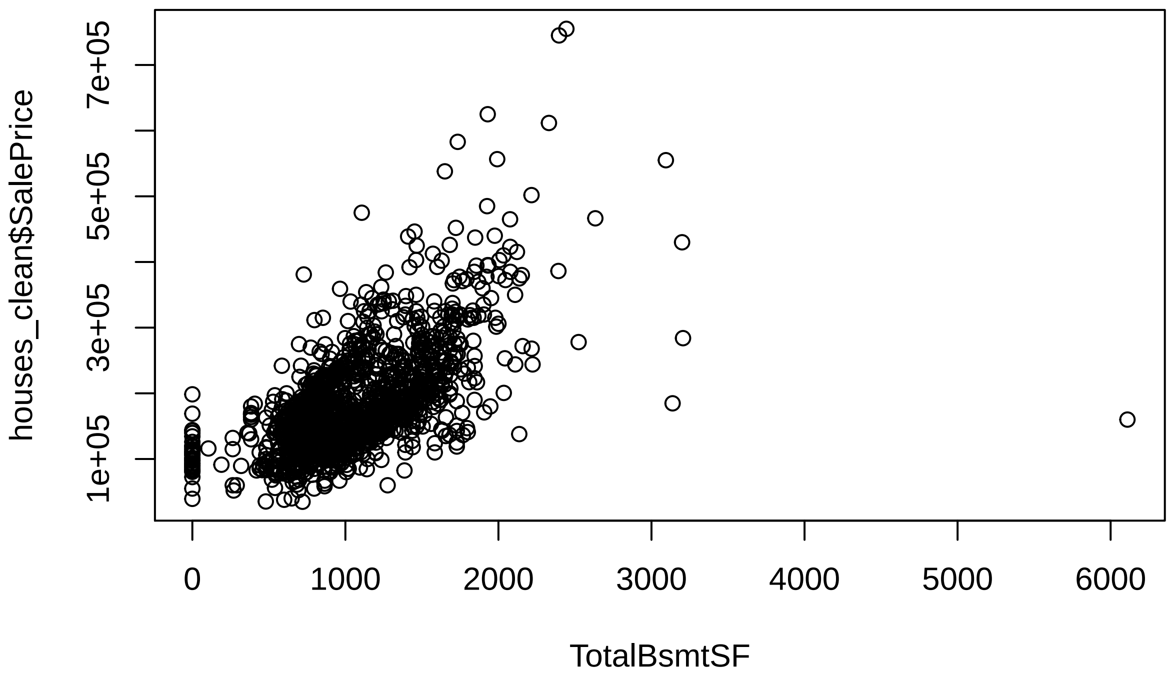

y
1e+05
3e+05
5e+05
7e+05
AsbShng
Brx
HdBoard
MetalSd
VinylSd
Exterior1st

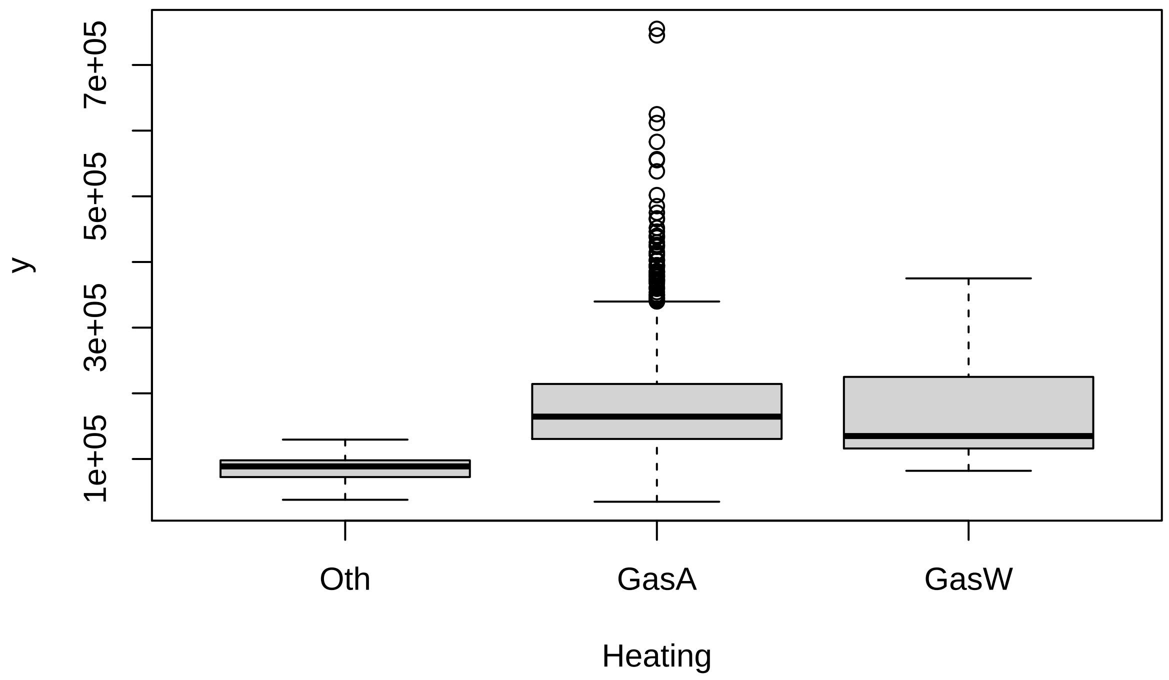


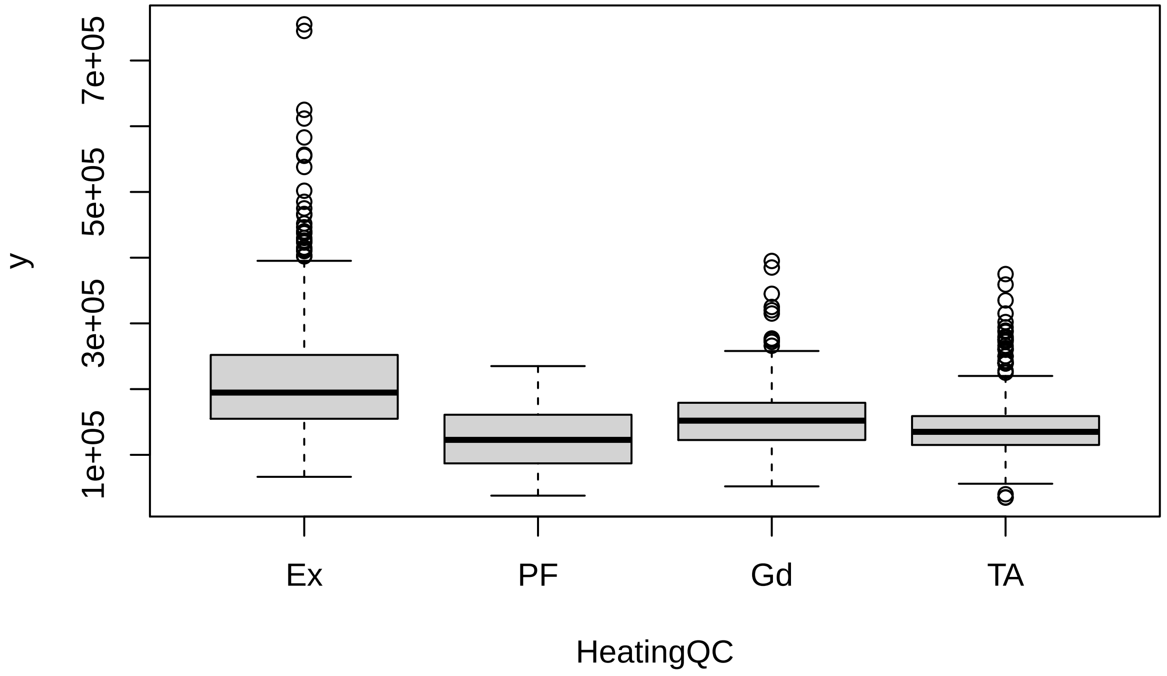
y
7e+05
5e+05
3e+05
1e+05
Ex
Fa
Gd
TA
ExterQual

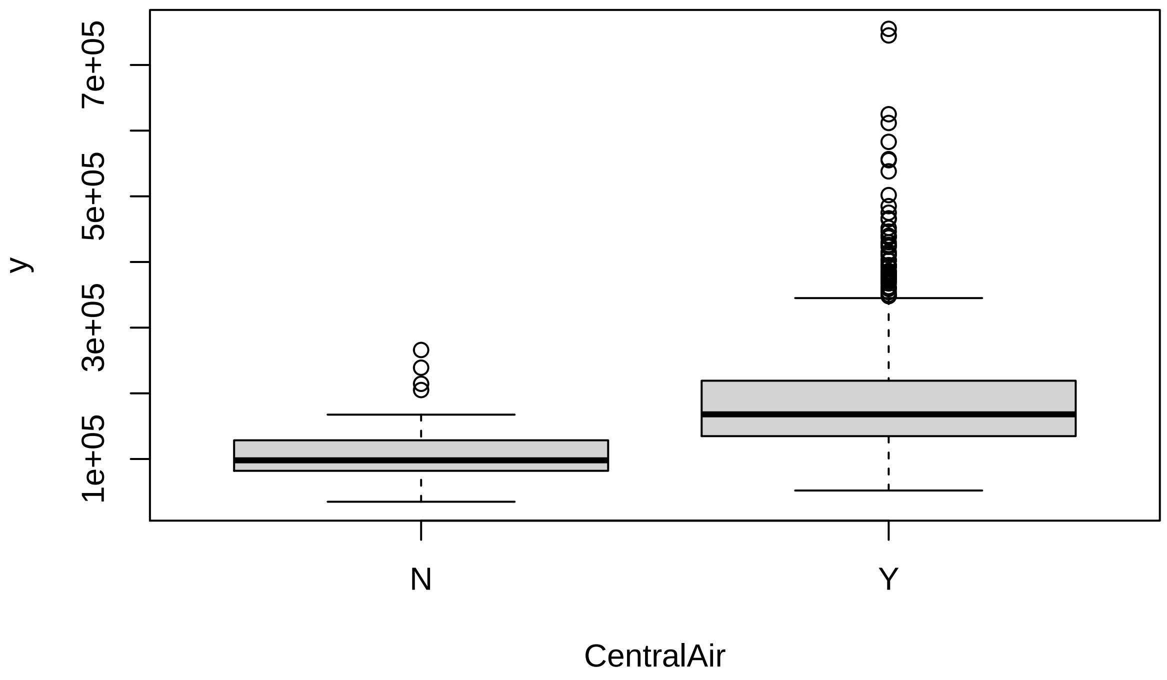

y
1e+05
3e+05
5e+05
7e+05
Ex
PF
Gd
TA
ExterCond

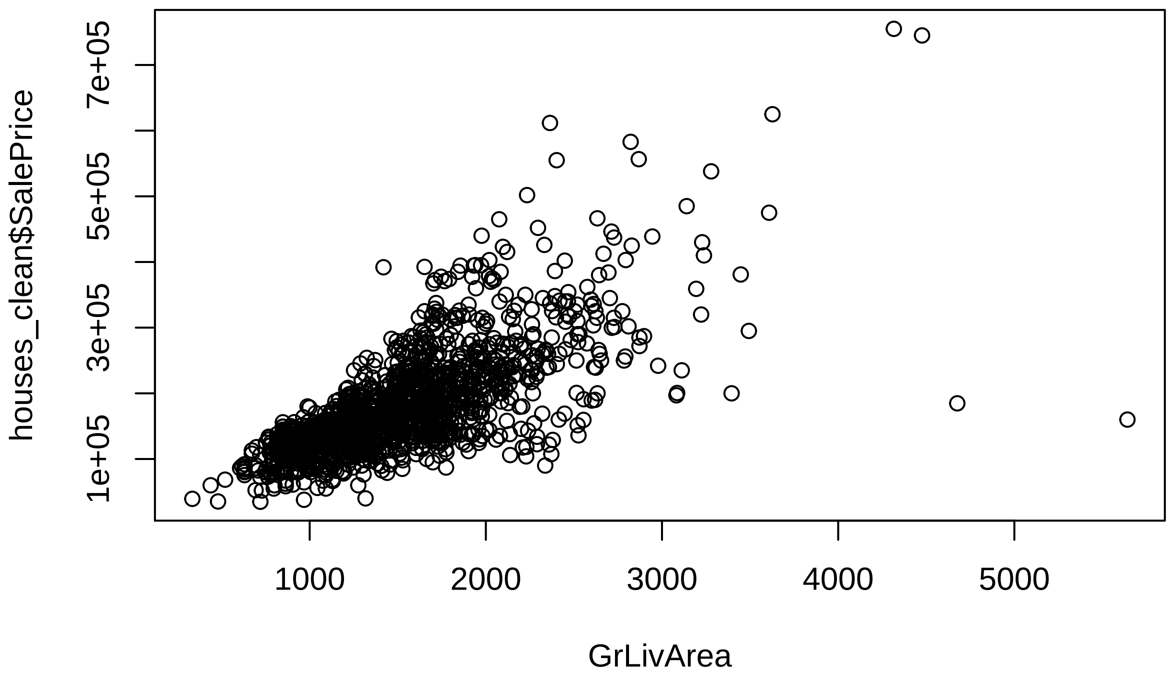

y
1e+05
3e+05
5e+05
7e+05
BrkTil
CBlock
PConc
Slab
Oth
Foundation

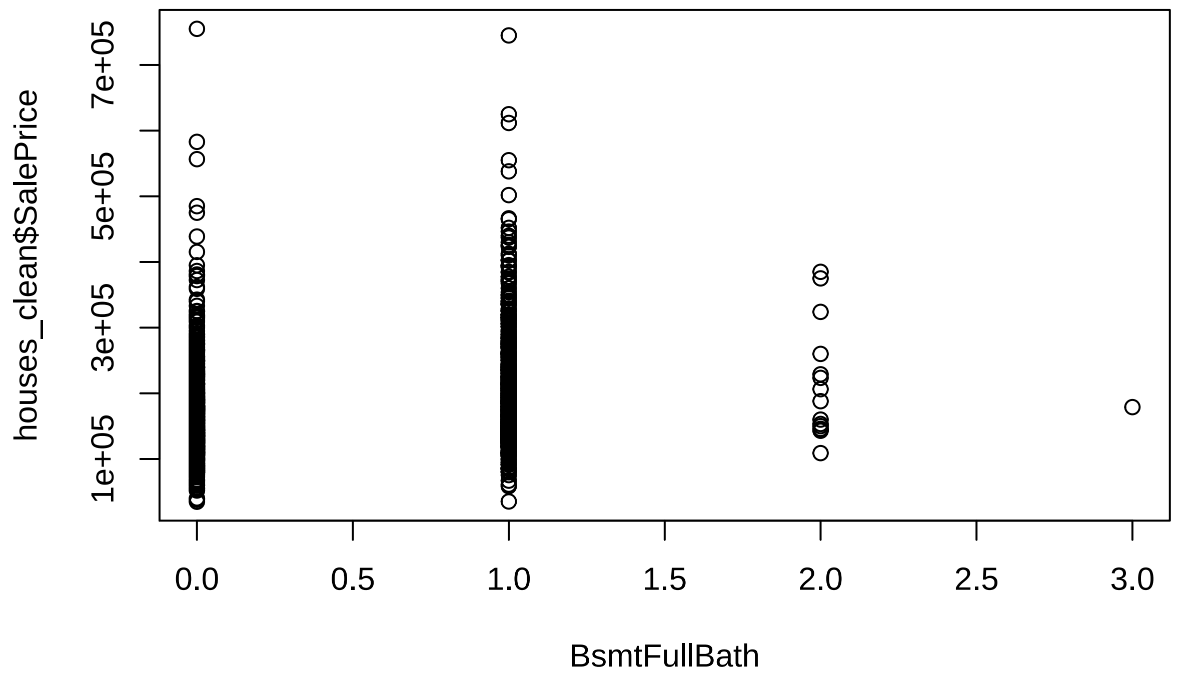

houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
0
1000
2000
3000
4000
5000
6000
TotalBsmtSF

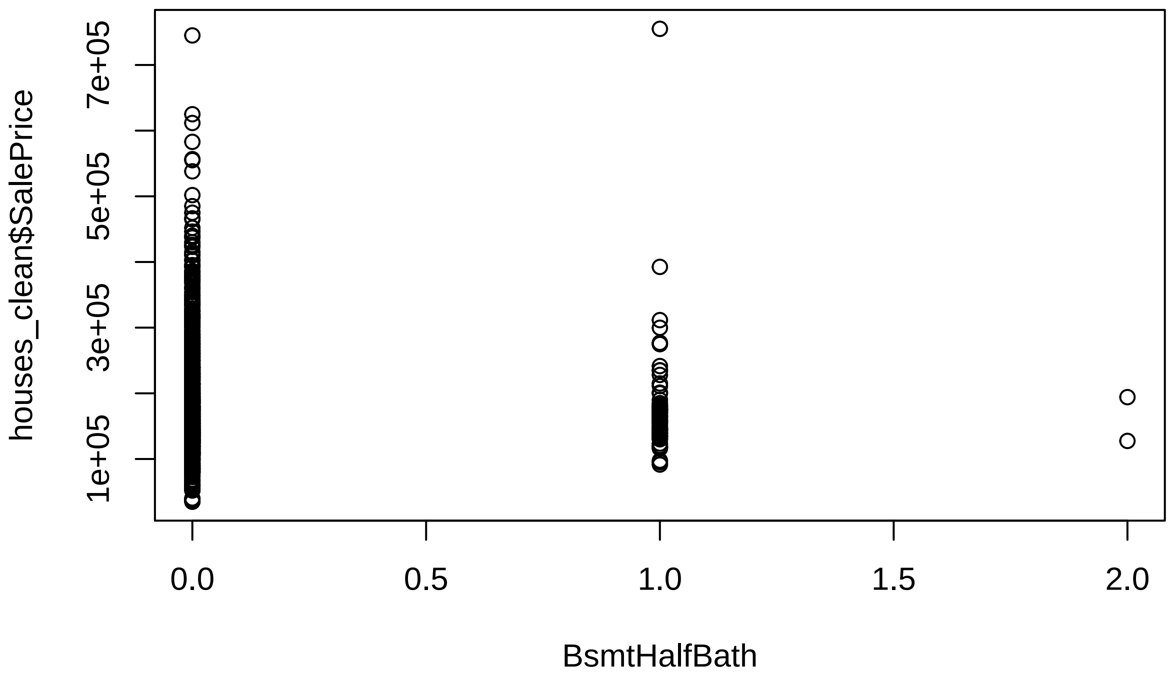

y
1e+05
3e+05
5e+05
7e+05
Oth
GasA
GasW
Heating

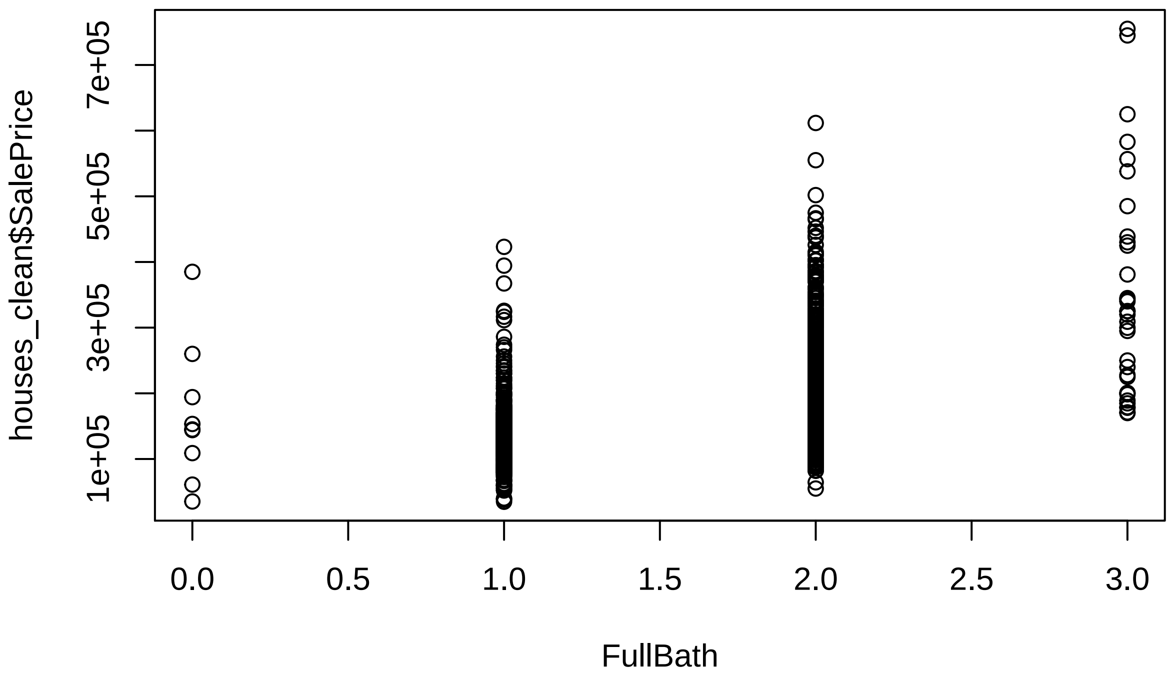

y
1e+05
3e+05
5e+05
7e+05
Ex
PF
Gd
TA
HeatingQC

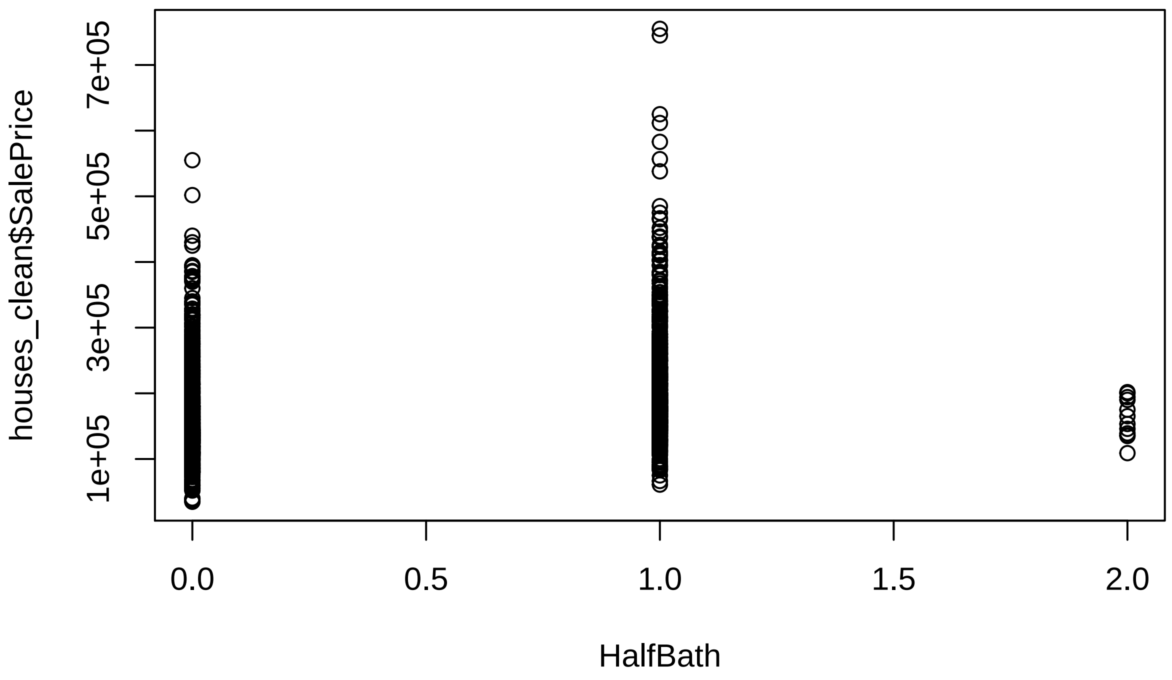

y
1e+05
3e+05
5e+05
7e+05
N
Y
CentralAir

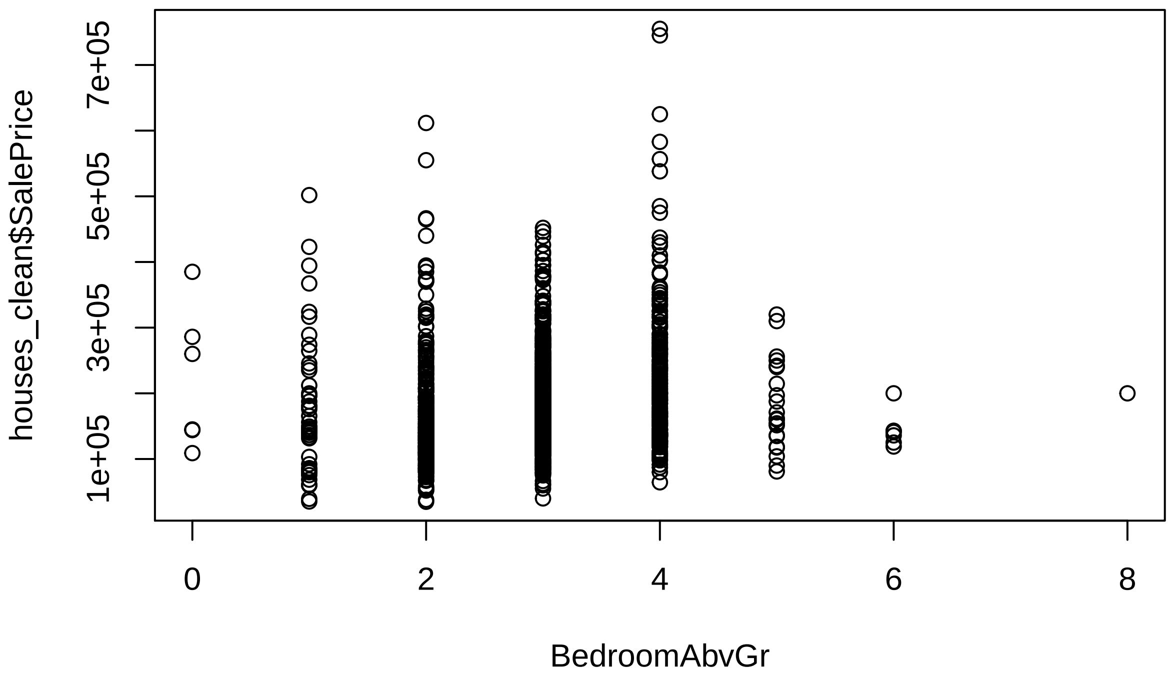

houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
1000
2000
3000
4000
5000
GrLivArea

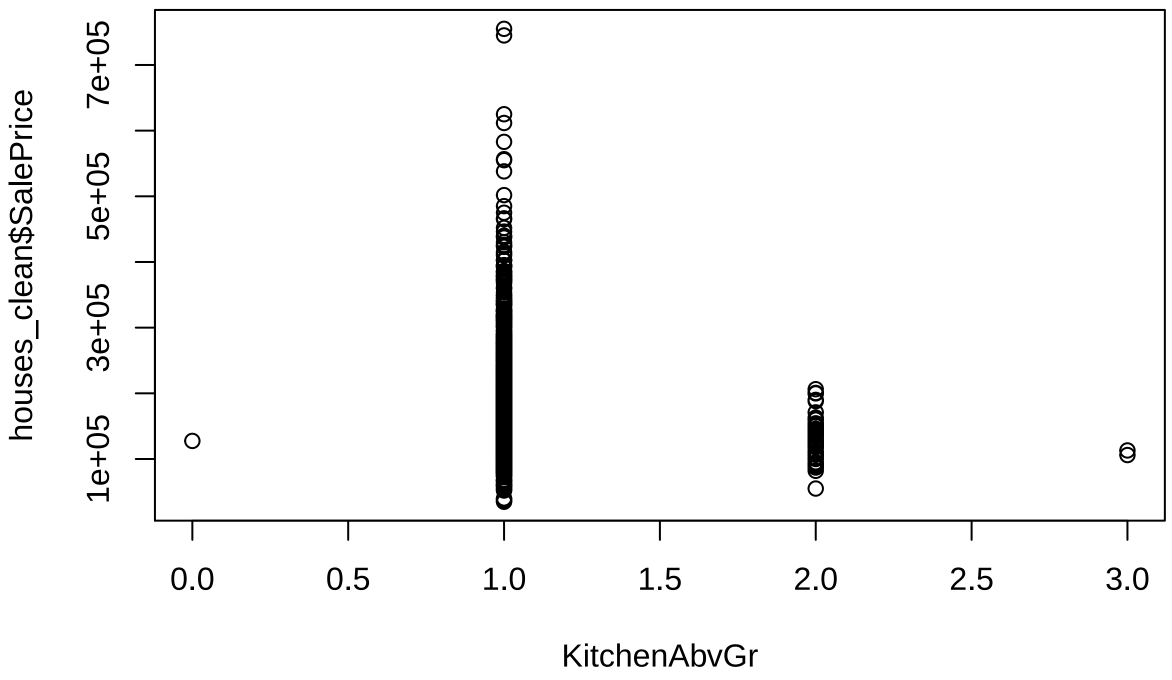

houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
0.0
0.5
1.0
1.5
2.0
2.5
3.0
BsmtFullBath

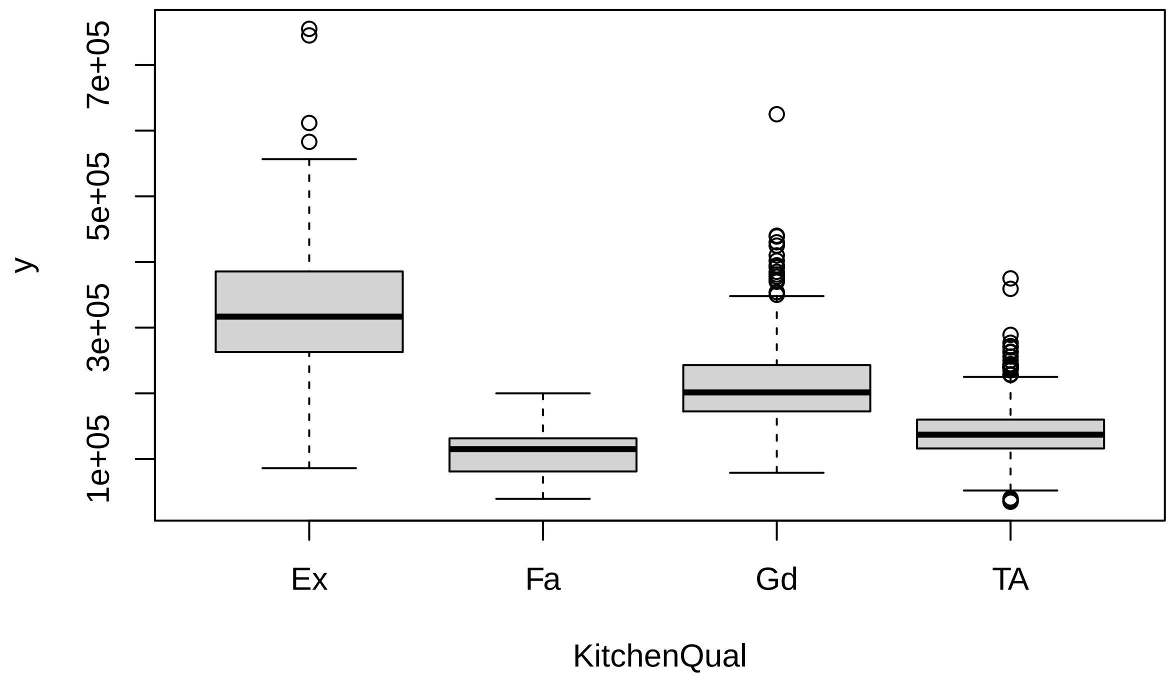

houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
0.0
0.5
1.0
1.5
2.0
BsmtHalfBath


houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
0.0
0.5
1.0
1.5
2.0
2.5
3.0
FullBath

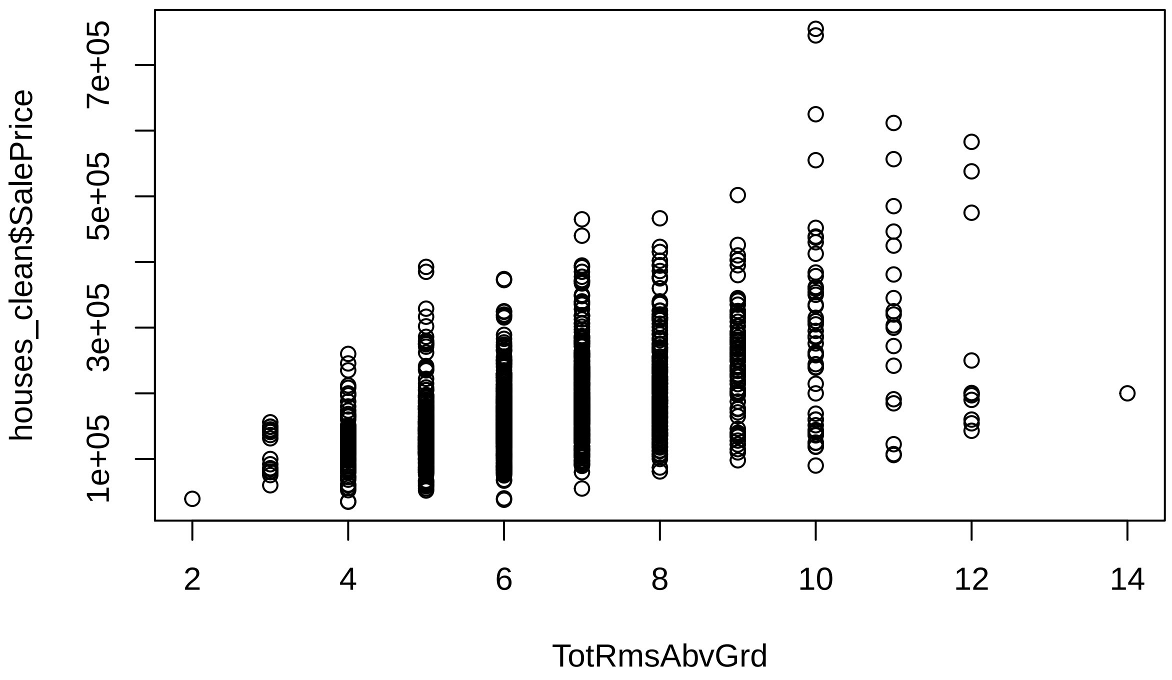

houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
0.0
0.5
1.0
1.5
2.0
HalfBath

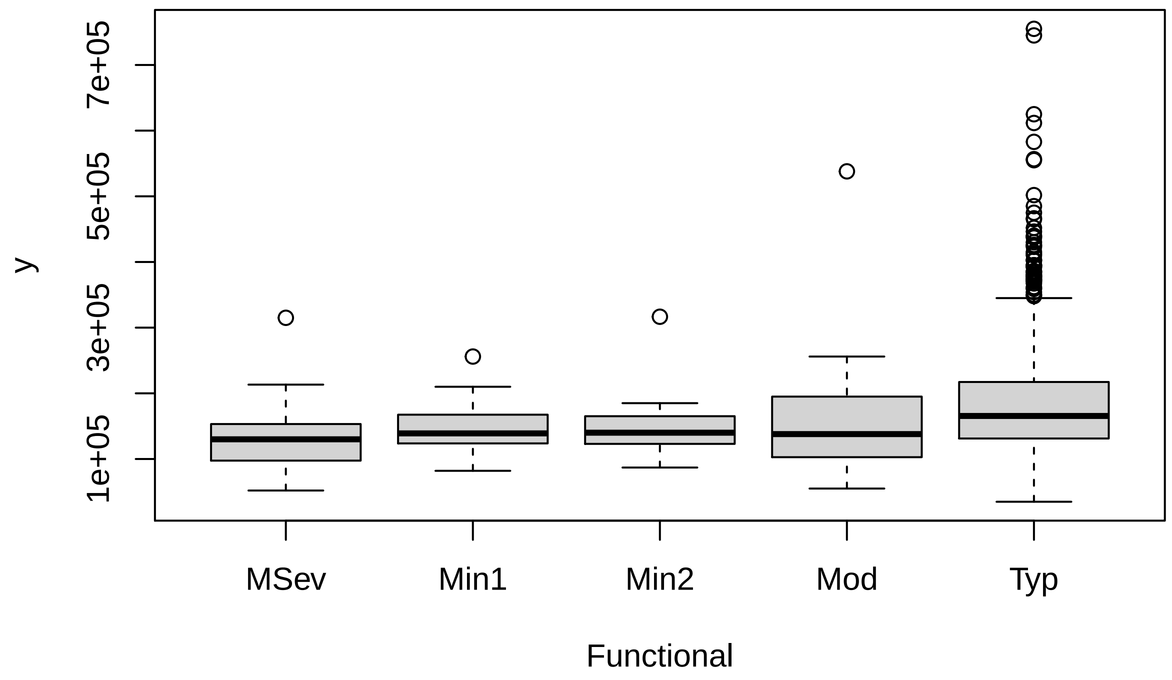

houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
0
2
4
6
8
BedroomAbvGr


houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
0.0
0.5
1.0
1.5
2.0
2.5
3.0
KitchenAbvGr

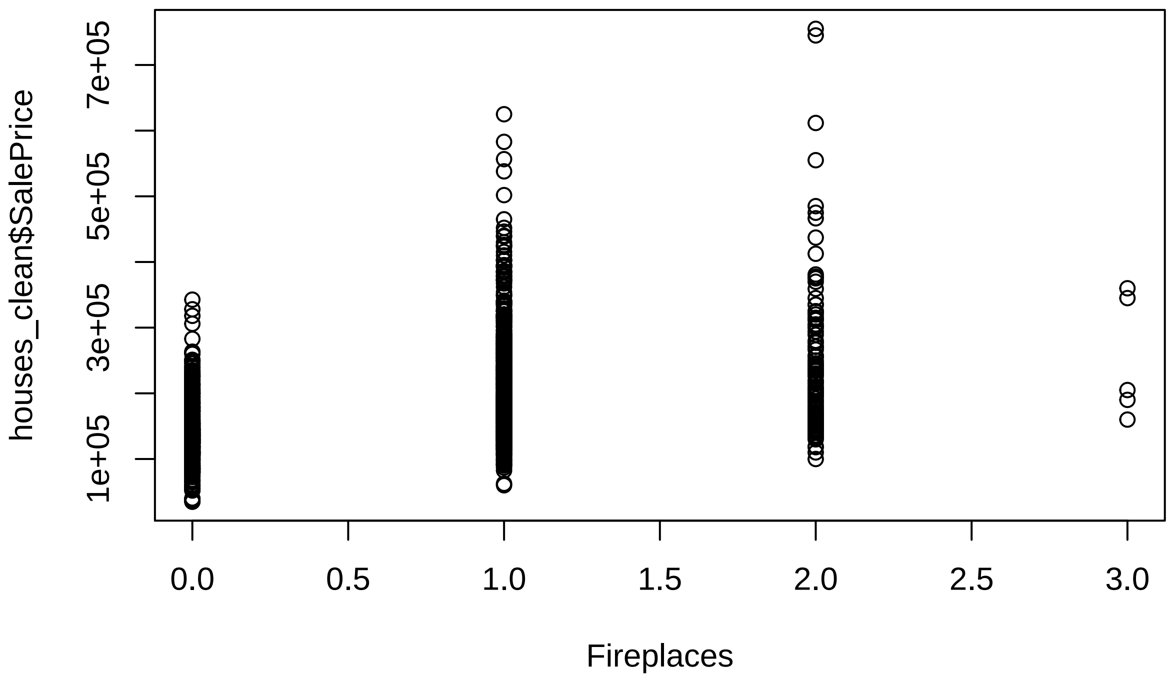


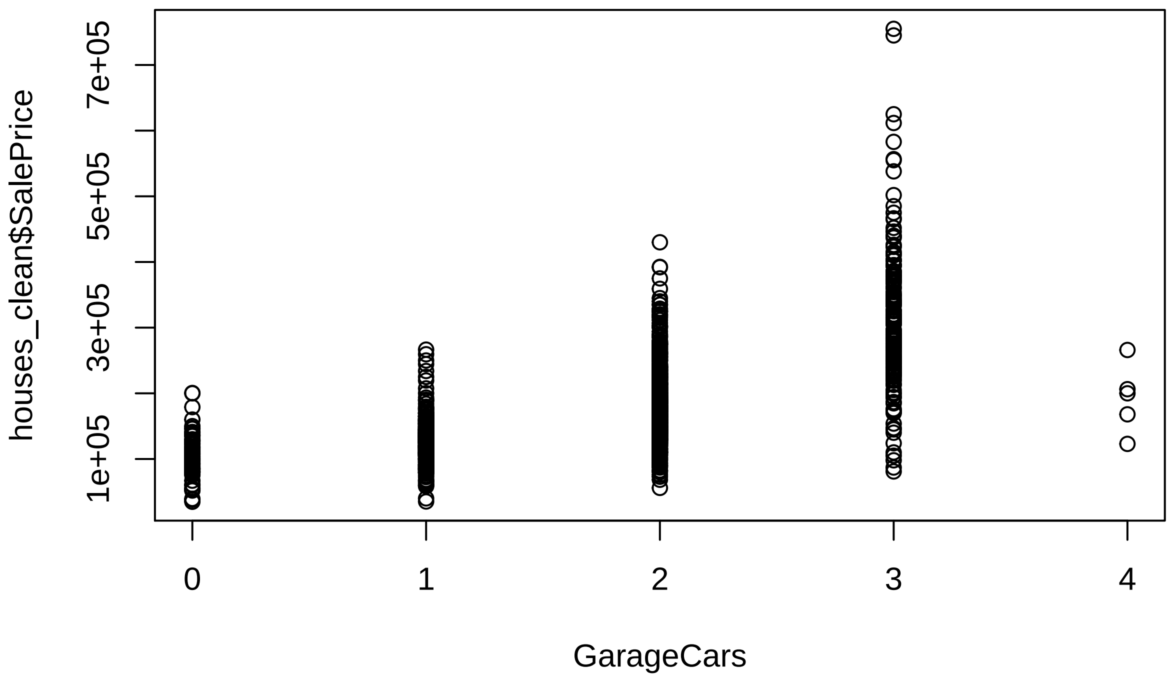
y
1e+05
3e+05
5e+05
7e+05
Ex
Fa
Gd
TA
KitchenQual

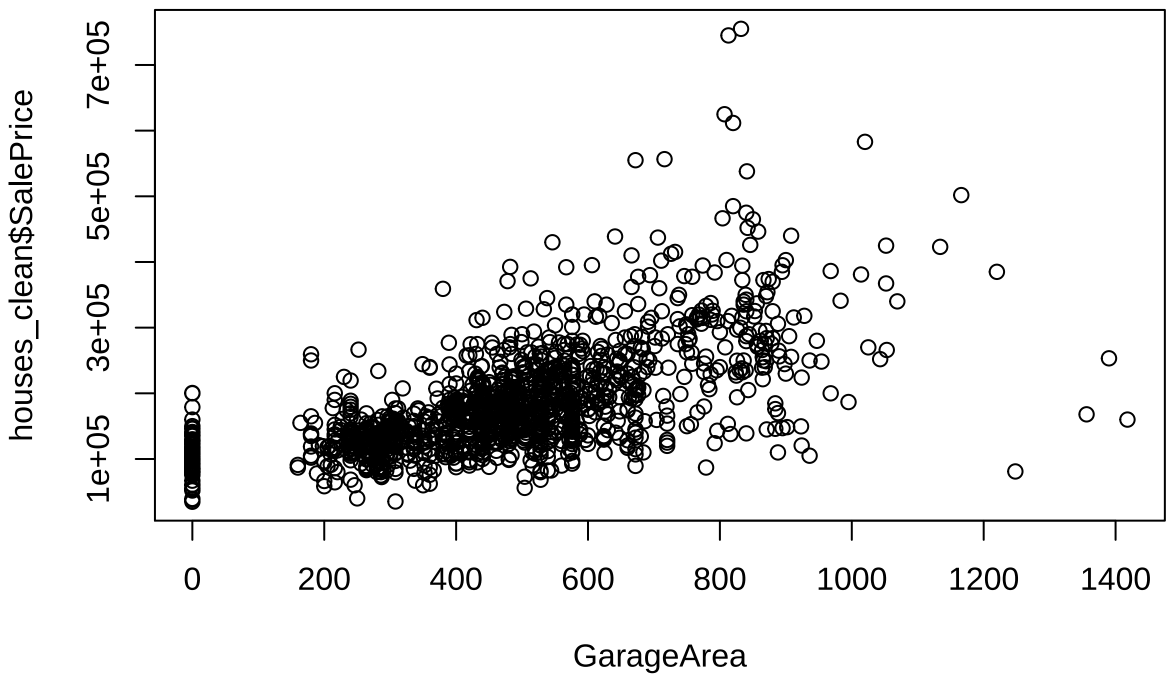

houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
2
4
6
8
10
12
14
TotRmsAbvGrd



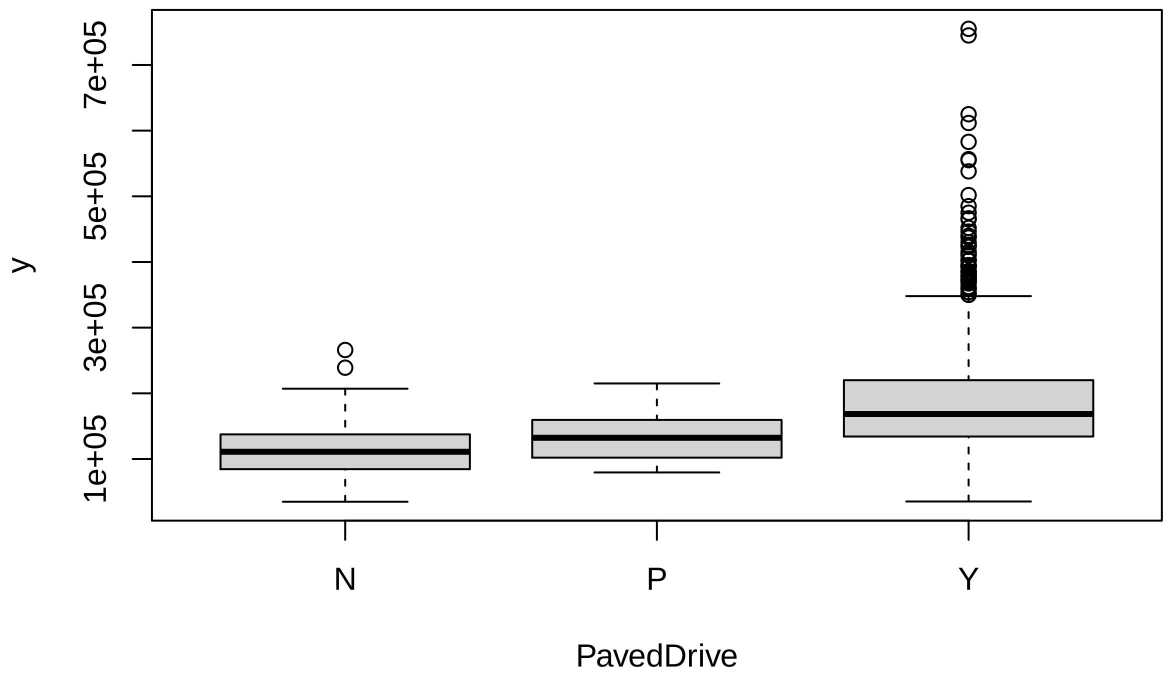
y
1e+05
3e+05
5e+05
7e+05
MSev
Min1
Min2
Mod
Typ
Functional

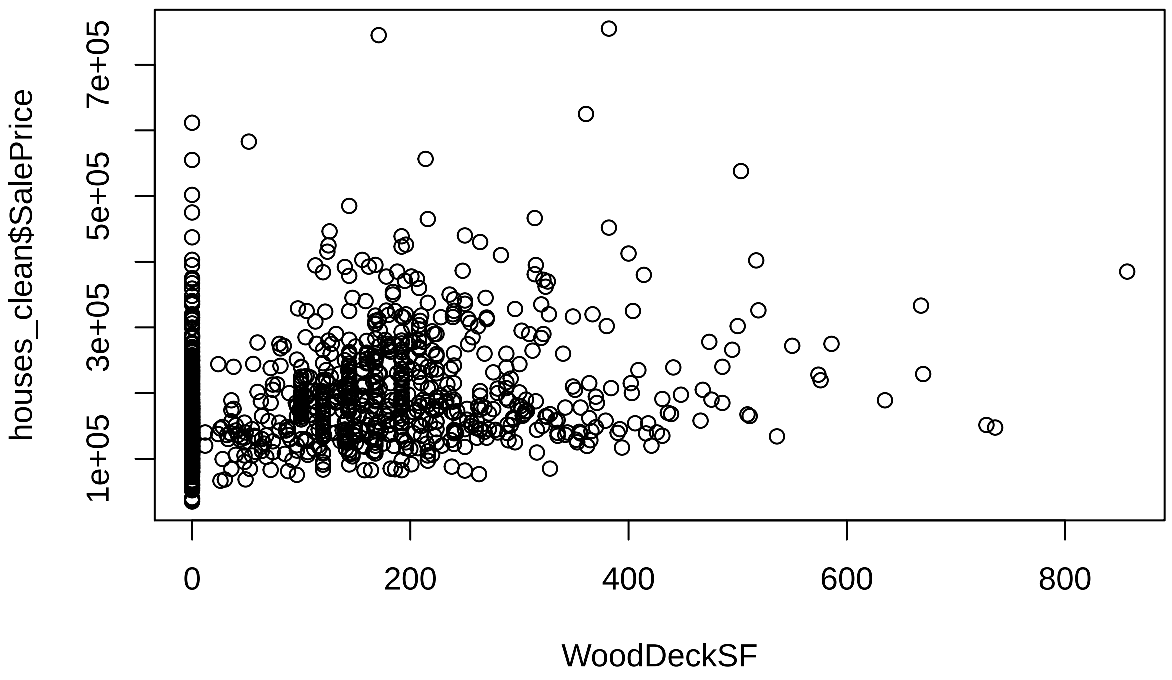

houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Fireplaces

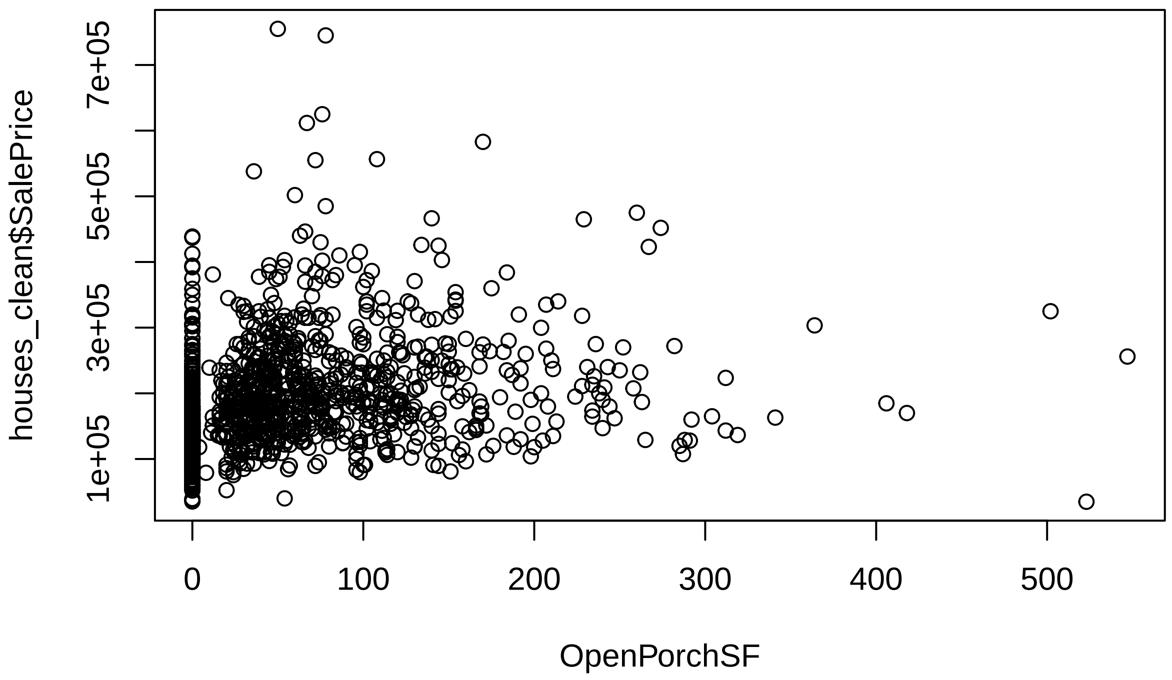

houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
0
1
2
3
4
GarageCars



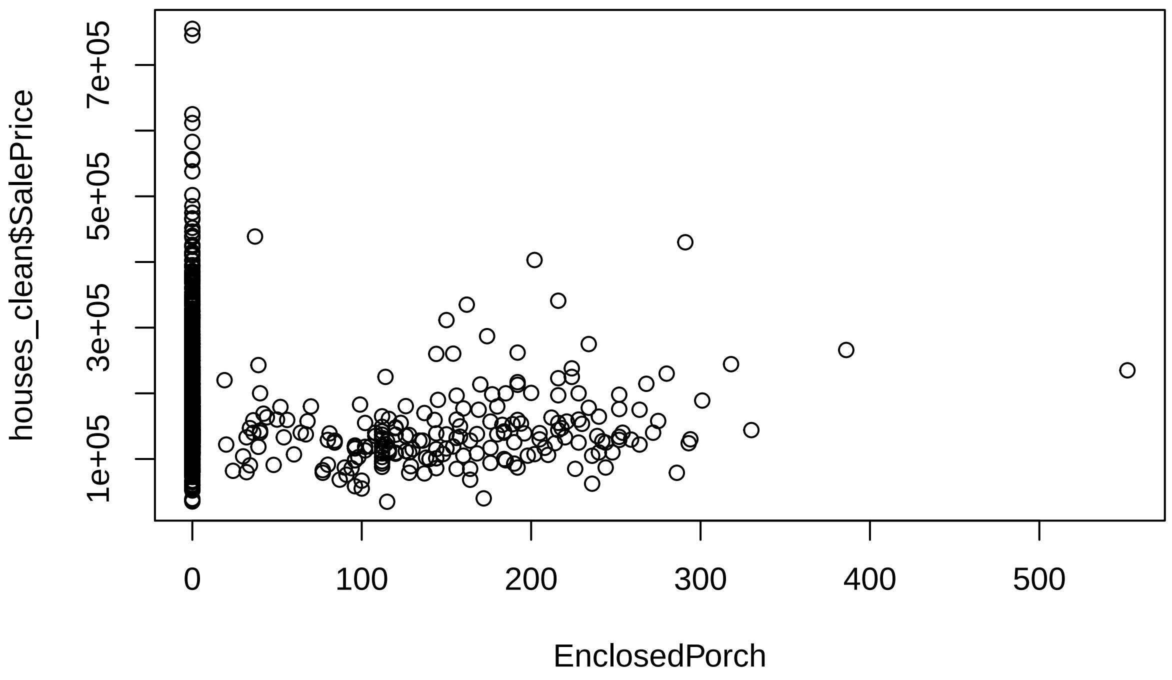
houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
0
200
400
600
800
1000
1200
1400
GarageArea


y
1e+05
3e+05
5e+05
7e+05
N
P
Y
PavedDrive

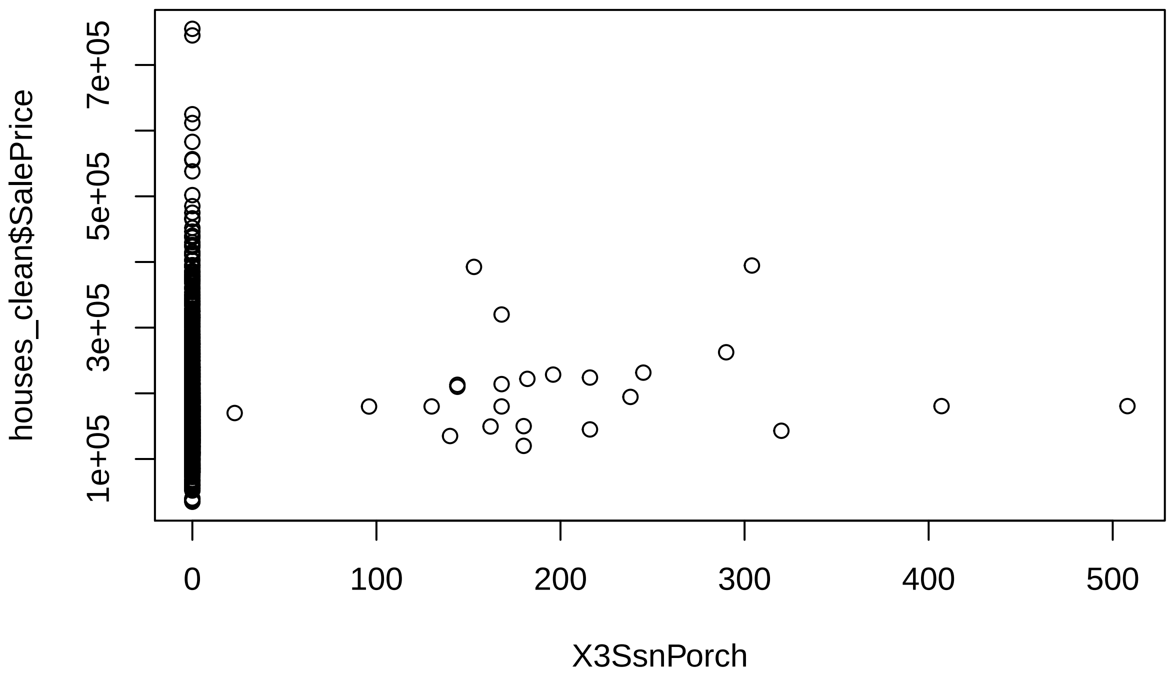

houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
0
200
400
600
800
WoodDeckSF



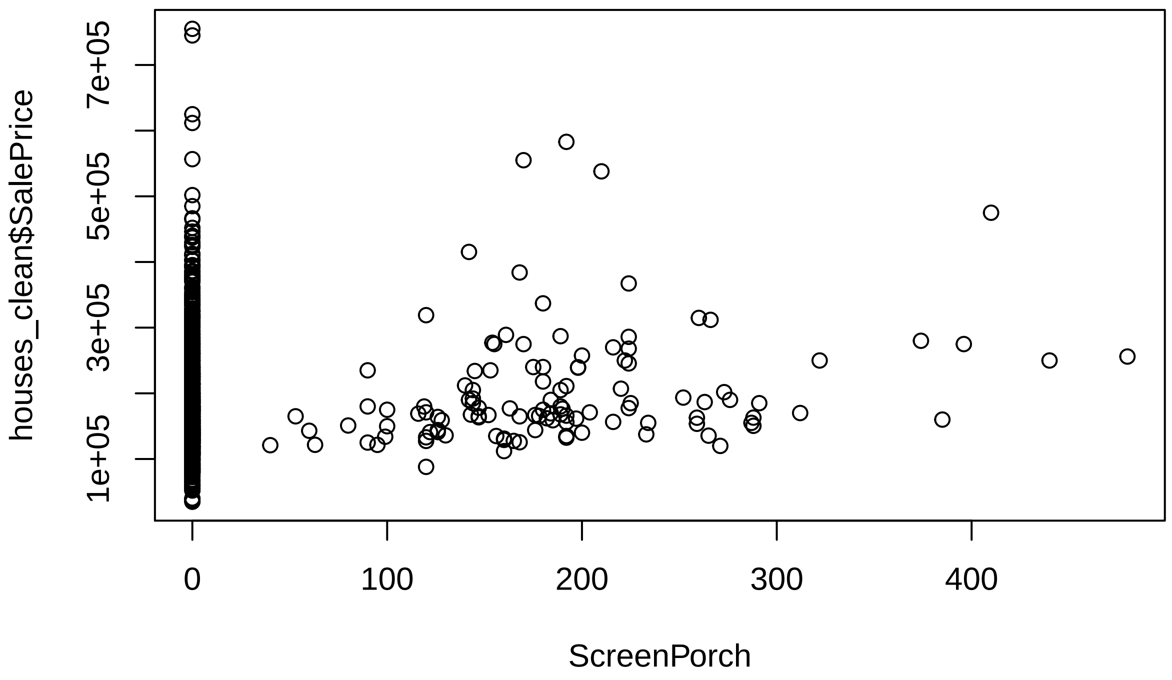
houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
0
100
200
300
400
500
OpenPorchSF

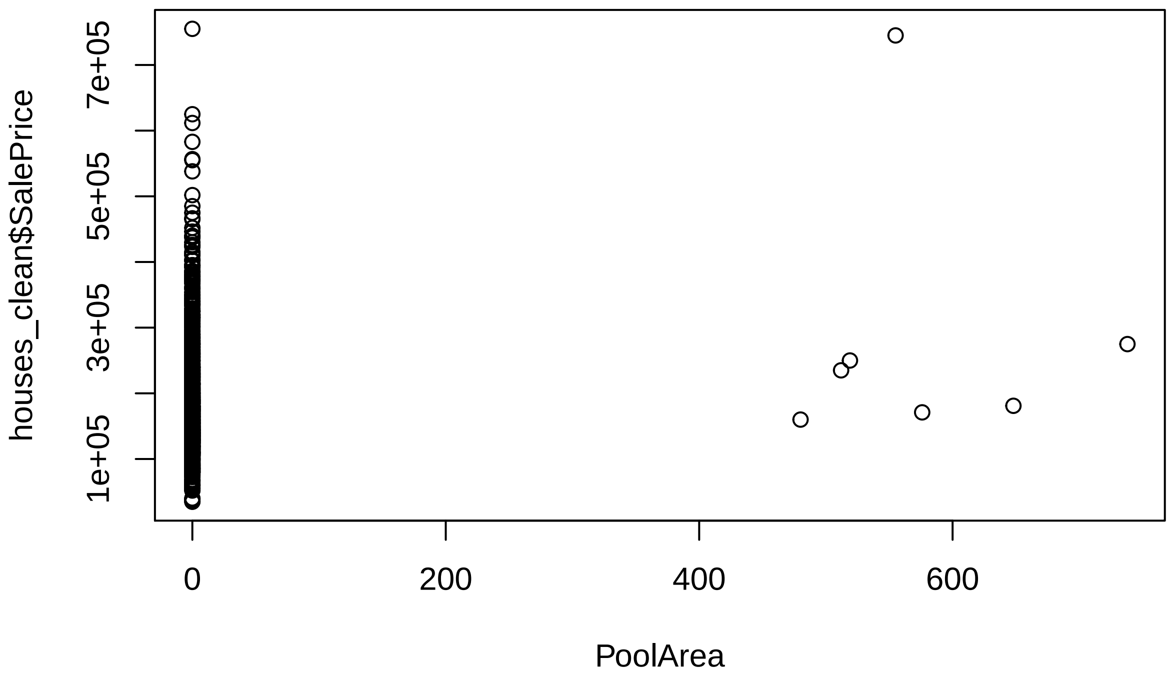


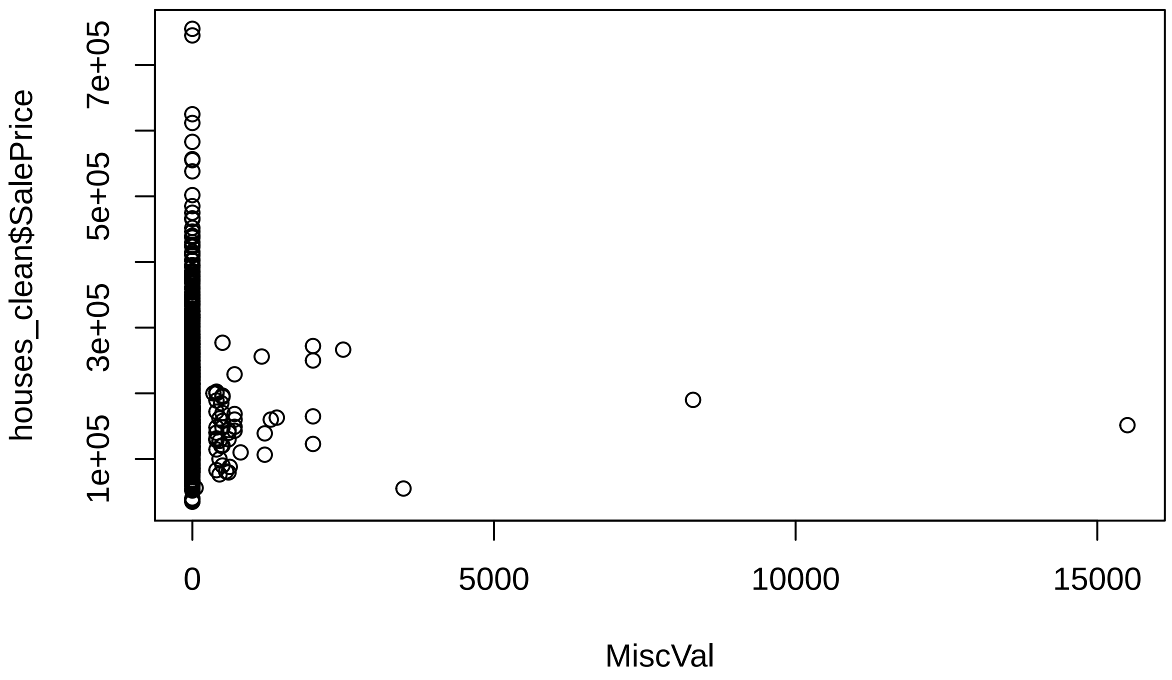
houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
0
100
200
300
400
500
EnclosedPorch

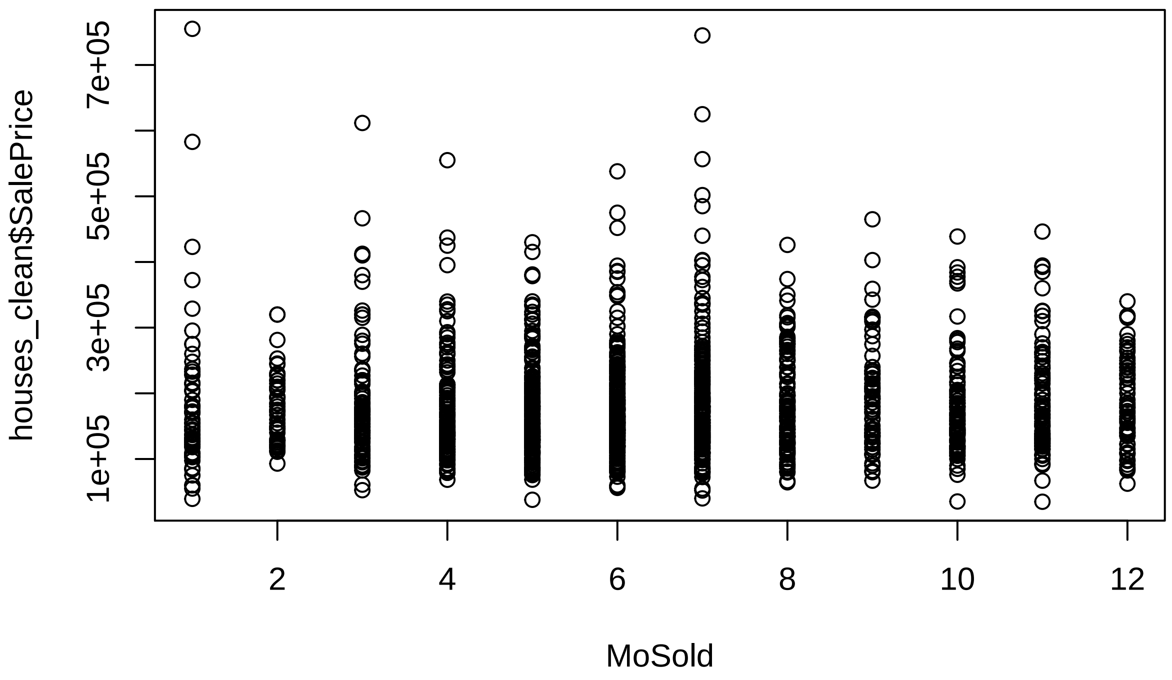


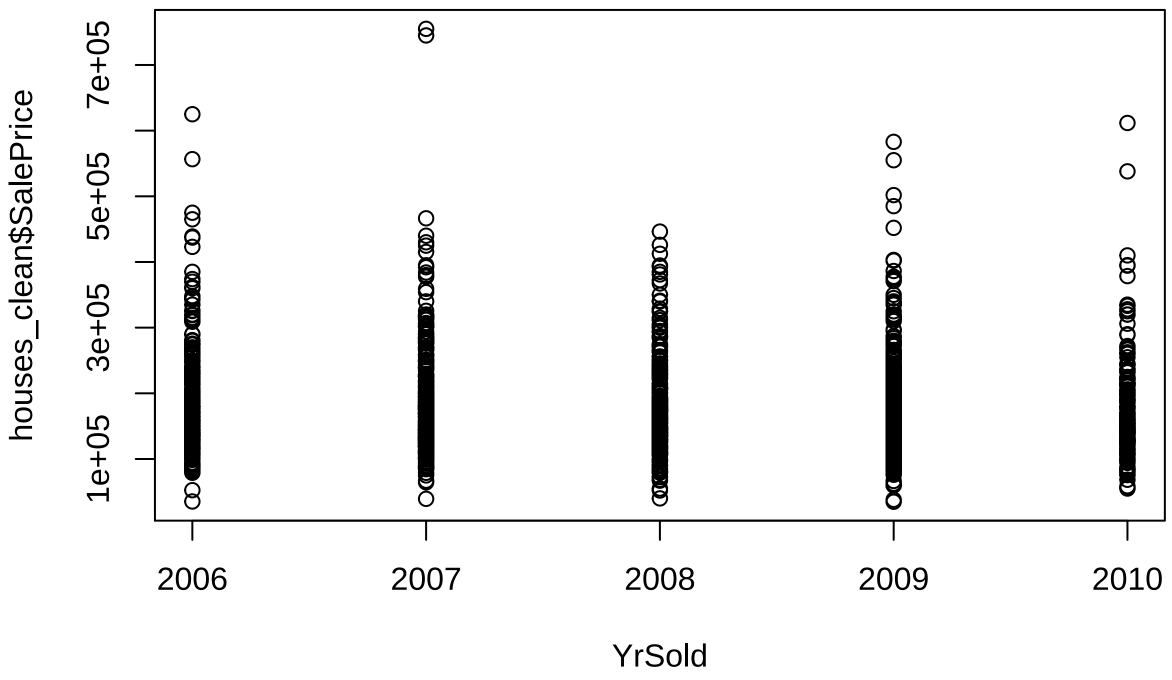
houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
0
100
200
300
400
500
X3SsnPorch

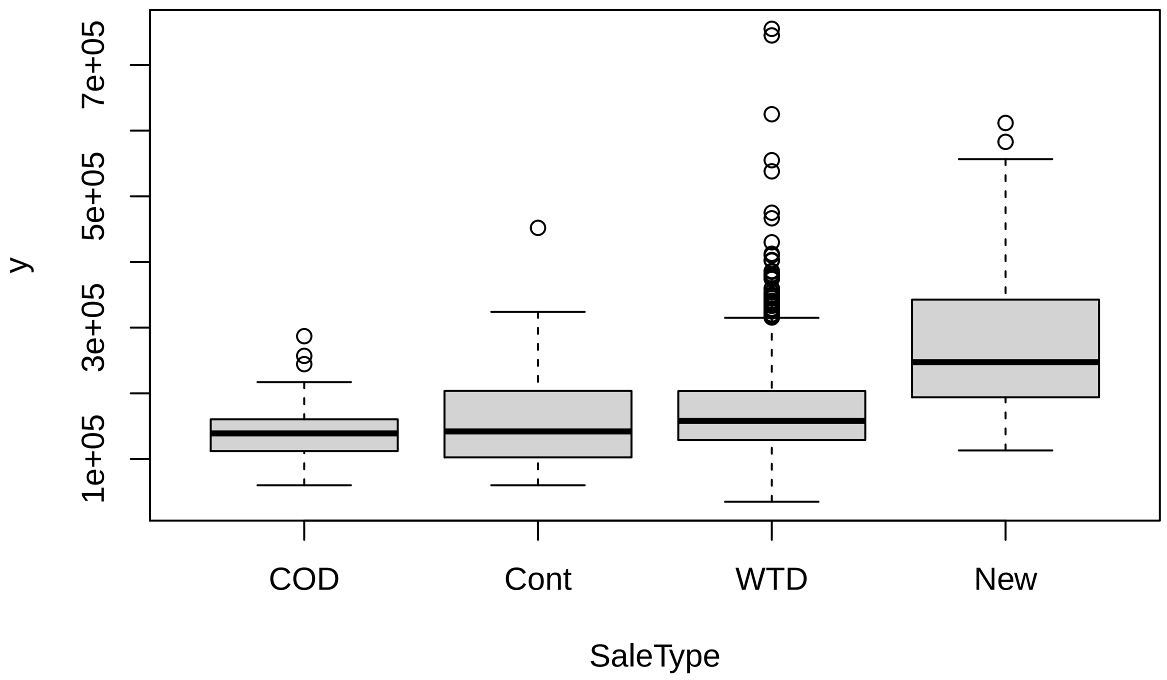

houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
0
100
200
300
400
ScreenPorch

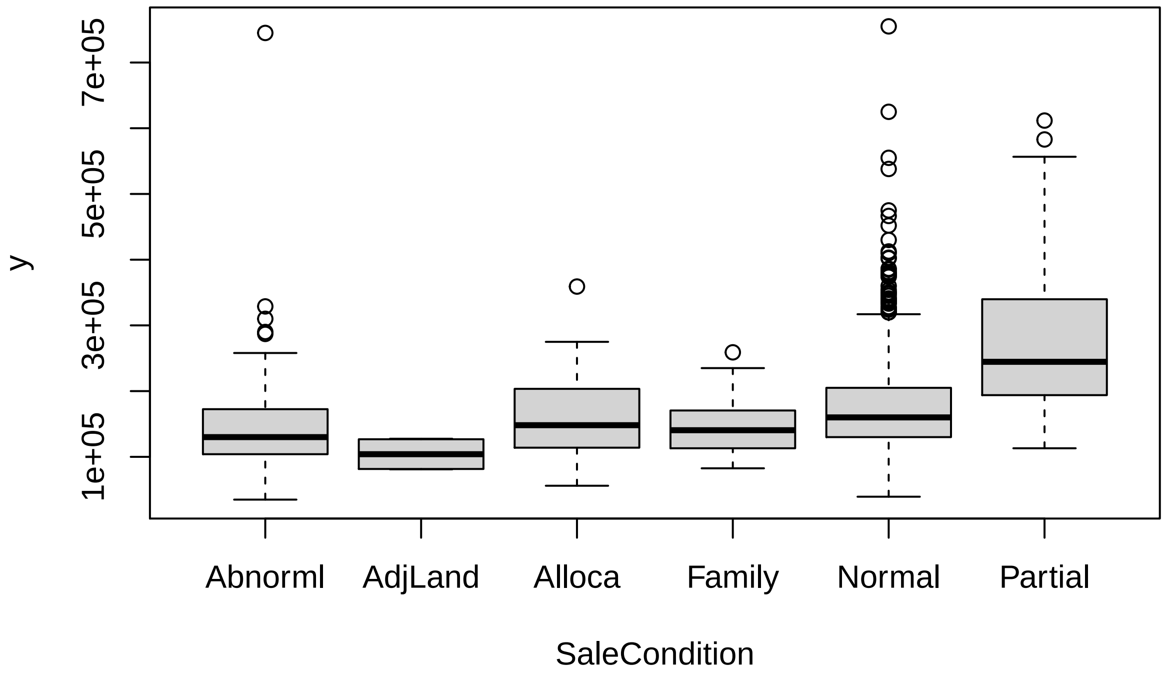

houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
0
200
400
600
PoolArea

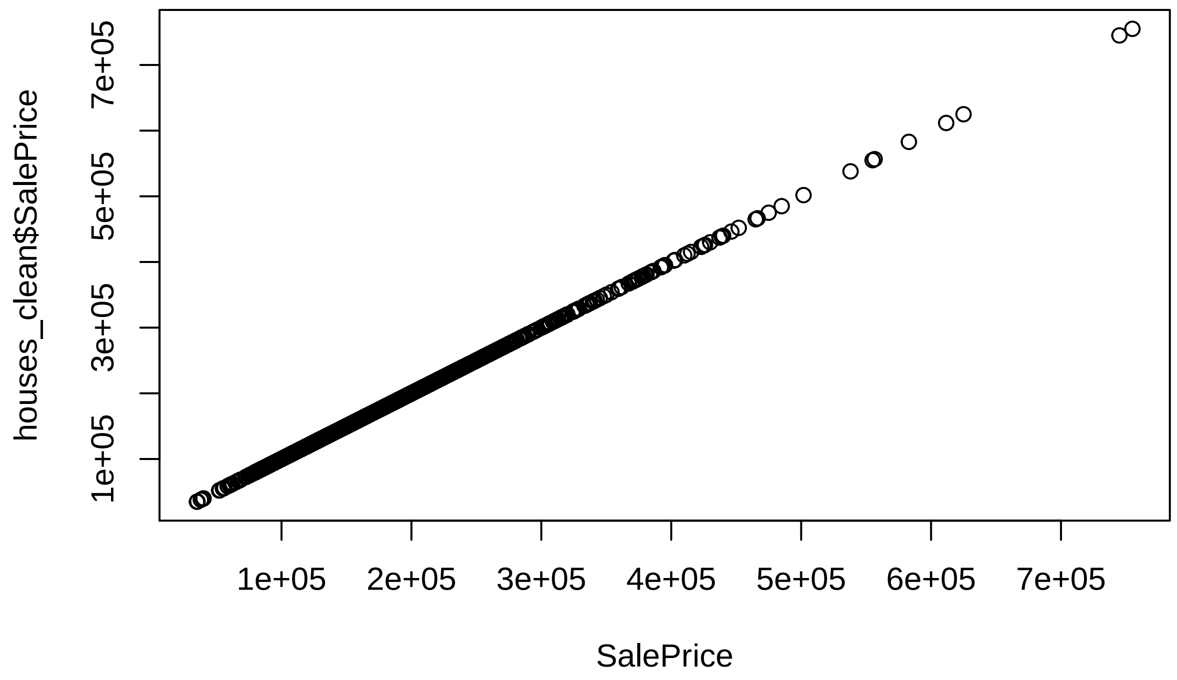

houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
0
5000
10000
15000
MiscVal


houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
2
4
6
8
10
12
MoSold


houses_clean$SalePrice
1e+05
3e+05
5e+05
7e+05
2006
2007
2008
2009
2010
YrSold


y
1e+05
3e+05
5e+05
7e+05
COD
Cont
WTD
New
SaleType


y
1e+05
3e+05
5e+05
7e+05
Abnorml
AdjLand
Alloca
Family
Normal
Partial
SaleCondition



Linear Model

```r
set.seed(3)
K=2
folds = sample(1:K,nrow(houses_clean),replace=T)
houses_lm<-list(NA)
pred_lm<-list(NA)
test_mse<-list(NA)
for(k in 1:K){
  CV.train = houses_clean[folds != k,]
  CV.test = houses_clean[folds == k,]
  CV.ts_y = CV.test$SalePrice
  houses_lm[[k]]=lm(SalePrice~.,data=CV.train)
  pred_lm[[k]]=predict(object=houses_lm[[k]], newdata=CV.test)
  test_mse[[k]]<-mean((pred_lm[[k]] - CV.ts_y)^2)
}
test_mse
```

```
## [[1]]
## [1] 5007538728
##
## [[2]]
## [1] 1980799459
```

```r
mean(c(test_mse[[1]], test_mse[[2]]))
```

```
## [1] 3494169093
```

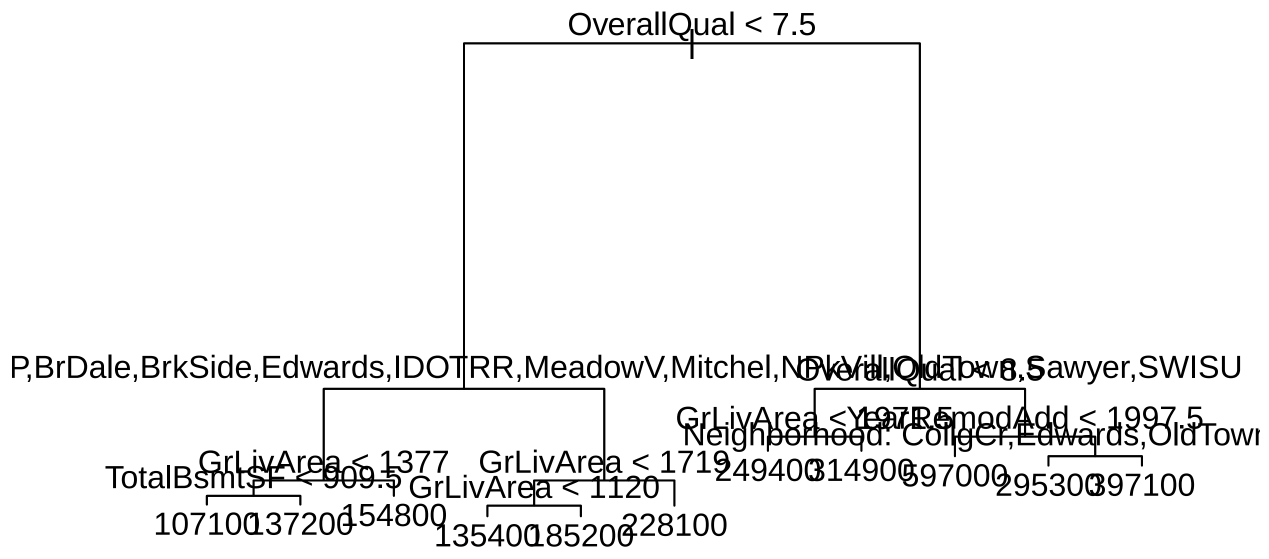
```r
lm2<-lm(SalePrice~., data=houses_clean)

predict_lm<-predict(lm2, newdata = houses_clean_tst)
```

Normal Regression Tree

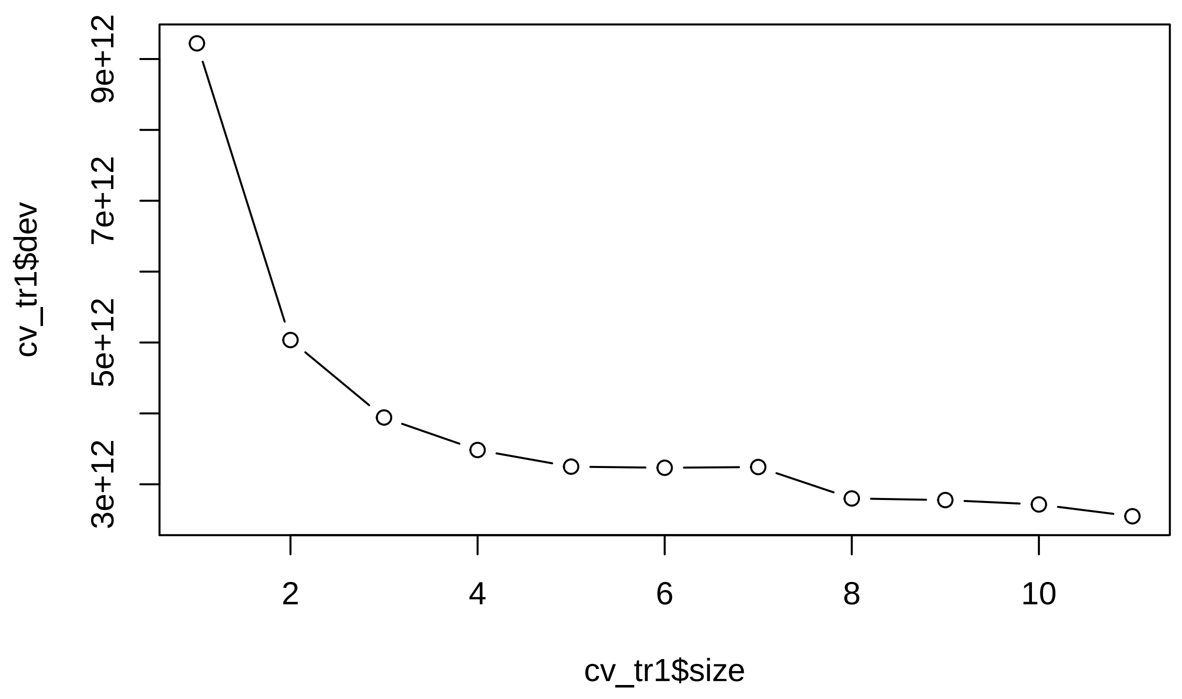
```r
##lets try a fresh stab using other methods... perhaps regression tree methods?
#ensure that we have factors instead of characters
tr1<-tree(SalePrice~., data=houses_clean)
summary(tr1)
```

```
##
## Regression tree:
## tree(formula = SalePrice ~ ., data = houses_clean)
## Variables actually used in tree construction:
## [1] "OverallQual"  "Neighborhood" "GrLivArea"    "TotalBsmtSF"  "YearRemodAdd"
## Number of terminal nodes:  11
## Residual mean deviance:  1.441e+09 = 2.088e+12 / 1449
## Distribution of residuals:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -212000  -20090   -1241       0   17780  223100
```

```r
#only 5 variables used here, overall qual, neighborhood, Grlivarea, total basementsf, year remmodel add
plot(tr1)
text(tr1, pretty=0)
```



```r
#try to prune or basic tree
cv_tr1<-cv.tree(tr1)
plot(cv_tr1$size ,cv_tr1$dev ,type="b")
```



```
#looks like our largest tree is the best, but perhaps a 8 node tree is comparable
#going to have to test predicted mse tho

K=2
folds = sample(1:K,nrow(houses_clean),replace=T)
houses_tr<-list(NA)
pred_tr<-list(NA)
test_mse<-list(NA)
for(k in 1:K){
  CV.train = houses_clean[folds != k,]
  CV.test = houses_clean[folds == k,]
  CV.ts_y = CV.test$SalePrice
  houses_tr[[k]]=tree(SalePrice~., data = CV.train)
  pred_tr[[k]]=predict(object=houses_tr[[k]], newdata=CV.test)
  test_mse[[k]]<-mean((pred_tr[[k]] - CV.ts_y)^2)
}
test_mse
```

```
## [[1]]
## [1] 1796069717
##
## [[2]]
## [1] 2035100373
```

```
mean(c(test_mse[[1]], test_mse[[2]]))
```

```
## [1] 1915585045
```

```r
predict_tr<-predict(tr1,newdata=houses_clean_tst)
```

Bagging

```r
tr2<-randomForest(SalePrice~., data=houses_clean, mtry = 54, importance = TRUE)

K=2
folds = sample(1:K,nrow(houses_clean),replace=T)
houses_bag<-list(NA)
pred_bag<-list(NA)
test_mse<-list(NA)
for(k in 1:K){
  CV.train = houses_clean[folds != k,]
  CV.test = houses_clean[folds == k,]
  CV.ts_y = CV.test$SalePrice
  houses_bag[[k]]=randomForest(SalePrice~., data = CV.train, mtry=52, importance = TRUE)
  pred_bag[[k]]=predict(object=houses_bag[[k]], newdata=CV.test)
  test_mse[[k]]<-mean((pred_bag[[k]] - CV.ts_y)^2)
}
test_mse
```

```
## [[1]]
## [1] 835608434
##
## [[2]]
## [1] 1171890774
```

```r
mean(c(test_mse[[1]], test_mse[[2]]))
```

```
## [1] 1003749604
```

Boosting

```r
#code for testing our boosting
houses_boost<-list(NA)
yhat.boost<-list(NA)
moderror<-list(NA)
bestmodel<-list(NA)
testn<-seq(from=100, to=10000, length.out = 10)
for(k in 1:K){
  CV.train = houses_clean[folds != k,]
  CV.test = houses_clean[folds == k,]
  CV.ts_y = CV.test$SalePrice
  for (i in 1:10){
    houses_boost[[i]]=gbm(SalePrice~.,data=CV.train, distribution="gaussian",n.trees=testn[[i]], interac
    yhat.boost[[i]]=predict(houses_boost[[i]] ,newdata =CV.test,n.trees=testn[[i]])
    test_mse[[i]]<-mean((yhat.boost[[i]] - CV.ts_y)^2)
  }
  moderror[[k]]<-test_mse[[which.min(test_mse)]]
  bestmodel[[k]]<-which.min(test_mse)}
moderror
```

```
## [[1]]
## [1] 906586511
##
## [[2]]
## [1] 1078534611
```

```
bestmodel
```

```
## [[1]]
## [1] 1
##
## [[2]]
## [1] 2
```

```
#best tree size is 1200
for(k in 1:K){
  CV.train = houses_clean[folds != k,]
  CV.test = houses_clean[folds == k,]
  CV.ts_y = CV.test$SalePrice
  for (i in 1:10){
    houses_boost[[i]]=gbm(SalePrice~.,data=CV.train, distribution="gaussian",n.trees=testn[[i]], intera
    yhat.boost[[i]]=predict(houses_boost[[i]] ,newdata =CV.test,n.trees=testn[[i]])
    test_mse[[i]]<-mean((yhat.boost[[i]] - CV.ts_y)^2)
  }
  moderror[[k]]<-test_mse[[which.min(test_mse)]]
  bestmodel[[k]]<-which.min(test_mse)}
#lets try it but w/ diff depths

for(k in 1:K){
  CV.train = houses_clean[folds != k,]
  CV.test = houses_clean[folds == k,]
  CV.ts_y = CV.test$SalePrice
  for (i in 1:5){
    houses_boost[[i]]=gbm(SalePrice~.,data=CV.train, distribution="gaussian",n.trees=1200, interaction.
    yhat.boost[[i]]=predict(houses_boost[[i]] ,newdata =CV.test,n.trees=1200)
    test_mse[[i]]<-mean((yhat.boost[[i]] - CV.ts_y)^2)
  }
  moderror[[k]]<-test_mse[[which.min(test_mse)]]
  bestmodel[[k]]<-which.min(test_mse)}

#our best ntrees is 1200, depth = 2
houses_boost=gbm(SalePrice~.,data=houses_clean, distribution="gaussian",n.trees=1200, interaction.depth
predict_boost=predict(houses_boost, newdata=houses_clean_tst)
```

```
## Using 1200 trees...
```

Conclusion

This assignment was very difficult. The largest component that I was not expecting to add large amounts of additional time and effort were the data cleaning and feature engineering components. I struggled a great deal with feature engineering for the large number of predictors. I would hope and assume that moving forward I would gain a better grasp on how to do practical feature engineering for extremely large datasets.